

# 7 Series FPGAs

# GTP Transceivers

## *User Guide*

UG482 (v1.8) November 19, 2014



#### Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at [www.xilinx.com/legal.htm#tos](http://www.xilinx.com/legal.htm#tos); IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at [www.xilinx.com/legal.htm#tos](http://www.xilinx.com/legal.htm#tos).

#### Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.

© Copyright 2012–2014 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.

## Revision History

The following table shows the revision history for this document.

Date	Version	Revision
01/03/2012	1.0	Initial Xilinx release.
02/21/2012	1.1	Changed "N" factor to "N1" and "N2" factors in <a href="#">Figure 2-10</a> , <a href="#">Equation 2-1</a> , and <a href="#">Table 2-7</a> . Revised <a href="#">Figure A-4</a> , <a href="#">Figure A-6</a> , <a href="#">Table B-1</a> , <a href="#">Table D-1</a> , and <a href="#">Table D-2</a> .
01/01/2012	1.1.1	Made typographical edits.
09/06/2012	1.2	Updated the second, third, and fourth paragraphs under <a href="#">Overview and Features in Chapter 1</a> . Updated description of PLL0_FBDIV/PLL1_FBDIV and added PLL0_FBDIV_45/ PLL0_FBDIV_45 attributes to <a href="#">Table 2-9</a> . Added <a href="#">Reset and Initialization</a> and <a href="#">Power Down in Chapter 2</a> . Updated Note 1 relevant to <a href="#">Figure 3-2</a> through <a href="#">Figure 3-5</a> . Updated descriptions of TXSTARTSEQ and GEARBOX_MODE attributes in <a href="#">Table 3-9</a> . Updated controller port clock domains and descriptions in <a href="#">Table 3-26</a> . Updated TXPI_SYN_FREQ_PPM[2:0] and TXPI_GREY_SEL attribute descriptions in <a href="#">Table 3-27</a> . Updated first introductory paragraph under <a href="#">TX Gearbox Operating Modes in Chapter 3</a> . Deleted <a href="#">Internal Sequence Counter Operating Mode</a> section in <a href="#">Chapter 3, Transmitter</a> . Added USE_PCS_CLK_PHASE_SEL and ES_CLK_PHASE_SE attributes to <a href="#">Table 4-20</a> . Added second and third paragraphs under <a href="#">Alignment Status Signals in Chapter 4</a> . Added last sentence to description of RXBYTEISALIGNED port in <a href="#">Table 4-25</a> . Added COMMA_ALIGN_LATENCY attribute to <a href="#">Table 4-26</a> . Updated description of GEARBOX_MODE attribute in <a href="#">Table 4-42</a> . Added <a href="#">Chapter 5, Board Design Guidelines</a> . Updated all package drawings in <a href="#">Appendix A, Placement Information by Package</a> . Updated <a href="#">Table B-1</a> .

Date	Version	Revision
10/23/2012	1.3	<p>Added Artix-7 device in <a href="#">Functional Description</a>, page 26, <a href="#">Single External Reference Clock Use Model</a>, page 33, and <a href="#">Multiple External Reference Clock Use Model</a>, page 34. Deleted XC7A350T in <a href="#">Figure 3-4</a> and <a href="#">Figure 3-5</a> footnotes. Deleted PCIe Protocol in <a href="#">Table 4-3</a>. Deleted XC7A350T devices in <a href="#">Table 5-2</a> and <a href="#">Figure 5-3</a>. Added additional ceramic filter capacitor to MGTAVCC_G[N] and MGTAVTT_G[N] pins in <a href="#">Table 5-14</a>. Deleted XC7A350T in <a href="#">Figure A-9</a>, <a href="#">Figure A-10</a>, <a href="#">Figure A-11</a>, <a href="#">Figure A-12</a>, <a href="#">Figure A-13</a>, and <a href="#">Figure A-14</a>. Deleted XC7A350T in <a href="#">Table B-1</a>.</p>
02/21/2013	1.4	<p>Replaced references to GTX transceiver with references to GTP transceiver throughout document.</p> <p>Chapter 2: Updated <a href="#">Figure 2-2</a>, <a href="#">Figure 2-12</a>, <a href="#">Figure 2-13</a>, <a href="#">Figure 2-14</a>, <a href="#">Figure 2-15</a>, <a href="#">Figure 2-16</a>, and <a href="#">Figure 2-17</a>. Updated <a href="#">Table 2-6</a>, rows one and two, and <a href="#">Table 2-8</a>, rows one and four. Revised <a href="#">Reset and Initialization</a>, last paragraph on <a href="#">page 41</a>. Updated <a href="#">Table 2-14</a>, rows five and six. Added <a href="#">Table 2-17</a>, and sections <a href="#">After Power-up and Configuration</a>, page 48, through <a href="#">TX Parallel Clock Source Reset</a>, page 49. Updated <a href="#">Table 2-18</a>, rows two, three, four, seven, twelve, thirteen, fifteen, and seventeen. Updated <a href="#">Figure 2-19</a> and added notes relevant to the figure. Updated <a href="#">Figure 2-20</a> and added notes relevant to the figure. Added <a href="#">GTP Transceiver RX PMA Reset</a>, page 58, including <a href="#">Figure 2-21</a> and notes relevant to the figure. Revised <a href="#">GTP Transceiver RX Component Resets</a>, page 58 by adding <a href="#">Table 2-22</a> and sections <a href="#">After Power-up and Configuration</a>, page 48 through <a href="#">After Comma Realignment</a>, page 63. Revised Loopback Functional description on <a href="#">page 67</a>. Updated <a href="#">Table 2-28</a>, row two. Updated <a href="#">Table 2-29</a>, rows three and seven and <a href="#">Table 2-30</a>, rows three and seven. Added <a href="#">Digital Monitor</a>, page 72 through <a href="#">page 75</a>.</p> <p>Chapter 3: Revised section <a href="#">TX Buffer Bypass</a>, page 97 through <a href="#">page 106</a>. Updated <a href="#">Figure 3-20</a>. Updated <a href="#">Table 3-24</a>, rows three and five, and <a href="#">Table 3-24</a>, row three.</p> <p>Chapter 4: Updated <a href="#">Table 4-3</a>, <a href="#">Table 4-4</a>, <a href="#">Table 4-5</a>, <a href="#">Table 4-6</a>, rows five and six, and <a href="#">Table 4-7</a>, row twelve. Added <a href="#">Use Mode</a>, page 135 through <a href="#">Figure 4-14</a>, <a href="#">page 141</a>. Added section <a href="#">Use Modes</a>, page 146 through <a href="#">Table 4-15</a>. Updated <a href="#">Figure 4-18</a>. Updated <a href="#">Table 4-17</a>, rows three and five. Added section <a href="#">Using RXRATE</a>, page 152 through <a href="#">page 153</a>. Revised section <a href="#">RX Buffer Bypass</a>, page 177 through <a href="#">page 190</a>. Updated <a href="#">Table 4-33</a> and <a href="#">Table 4-33</a>, rows five and ten.</p> <p>Chapter 5: Updated <a href="#">Table 5-2</a>, rows one and two, and <a href="#">Table 5-11</a>, rows three and four.</p> <p>Appendix A: Updated <a href="#">Figure A-4</a> through <a href="#">Figure A-14</a>.</p> <p>Appendix B: Updated <a href="#">Table B-1</a>.</p>
04/15/2013	1.5	<p>Added last two rows in <a href="#">Table 2-22</a>. Added three sentences to <a href="#">Loopback Functional Description</a>, page 26. Changed “DEN” to “DRPEN” in <a href="#">Table 2-29</a> and <a href="#">Table 2-30</a>. Added a note to <a href="#">Figure 2-23</a> and <a href="#">Figure 2-24</a>. Revised <a href="#">TX Buffer Bypass Functional Description</a>, page 95 and <a href="#">Table 3-15</a>. Revised <a href="#">TX Buffer Bypass Use Modes</a>, page 100, deleted <a href="#">Figure 3-12</a>, <a href="#">TX Buffer Bypass, Single lane Auto mode Port Connection</a>, and replaced <a href="#">Figure 3-12</a> and notes relevant to it. Revised <a href="#">Using TX Buffer Bypass in Multi-Lane Mode</a>, page 102 (and removed “Manual” from section title and text). Deleted section titled “Using TX Buffer Bypass in Multi Lane Auto Mode.” Added last two rows to <a href="#">Table 4-2</a>. Changed “INCP” to “IPCM” in <a href="#">Table 4-3</a>, <a href="#">Table 4-4</a>, and <a href="#">Table 4-5</a>. Changed RXCDR_CFG attribute type from 72- to 83-bit hex in <a href="#">Table 4-12</a>.</p>
08/28/2013	1.6	<p>Added devices XC7A35T-CSG325 (Preliminary), XC7A35T-FGG484 (Preliminary), XC7A50T-CSG325 (Preliminary), XC7A50T-FGG484 (Preliminary), XC7A75T-FGG484, and XC7A75T-FGG676.</p>

Date	Version	Revision
04/03/2014	1.7	<p>Added devices XC7A35T-CPG236, XC7A50T-CPG236, and XC7Z015-CLG485. Changed SIM_VERSION type from "Real" to "String" in <a href="#">Table 1-3</a>. Changed RX rate change from "RX PCS" to "Entire RX" in <a href="#">Table 2-22</a>. Expanded <a href="#">RX Rate Change, page 62</a>. Expanded descriptions for DRPEN in <a href="#">Table 2-29</a> and <a href="#">Table 2-30</a>. Modified the descriptions for RXOSCALRESET through RXOSINTDONE in <a href="#">Table 4-11</a>. Changed direction of RXCHARISK[3:0] from "In" to "Out" in <a href="#">Table 4-27</a>. Updated <a href="#">Table 5-2</a> and <a href="#">Figure 5-3</a> for new devices/packages. Added <a href="#">Table 5-3</a>, <a href="#">Table 5-8</a>, <a href="#">Table 5-9</a>, and <a href="#">Table 5-10</a>, for new devices/packages. Expanded <a href="#">SelectIO Usage Guidelines, page 241</a>. Added CPG236, CSG325, and CLG485 package placement diagrams (<a href="#">Figure A-1</a>, <a href="#">Figure A-2</a>, and <a href="#">Figure A-3</a>). Updated <a href="#">Figure A-4</a> for new devices. Updated <a href="#">Table B-1</a> and added <a href="#">Table B-2</a> for new devices.</p>
11/19/2014	1.8	<p>Added device XC7A15T (-PG236, -CPG236, and -CLG485 packages). Enhanced descriptions for port O and ODIV2 in <a href="#">Table 2-1</a>. Added ports BGBYPASSB, BGMONITORENB, BGPDB, BGRCALOVRD, and RCALENB to <a href="#">Table 2-8</a>. Added last two rows to <a href="#">Table 2-17</a>. Added second paragraph under <a href="#">PLL Power Down, page 66</a>. Updated digital monitor <a href="#">Functional Description, page 72</a> and added ports DMONITORCLK and DMONFIFORESET to <a href="#">Table 2-31</a>.</p>

# *Table of Contents*

---

Revision History .....	2
<b>Preface: About This Guide</b>	
Guide Contents .....	9
Additional Resources .....	10
Additional References.....	10
<b>Chapter 1: Transceiver and Tool Overview</b>	
Overview and Features .....	11
7 Series FPGAs Transceivers Wizard.....	16
Simulation.....	16
Implementation.....	20
<b>Chapter 2: Shared Features</b>	
Reference Clock Input Structure.....	23
Reference Clock Selection and Distribution .....	26
PLL .....	35
Reset and Initialization.....	39
Power Down .....	63
Loopback .....	66
Dynamic Reconfiguration Port .....	69
Digital Monitor .....	72
<b>Chapter 3: Transmitter</b>	
TX Overview.....	77
FPGA TX Interface .....	78
TX 8B/10B Encoder.....	85
TX Gearbox .....	88
TX Buffer .....	95
TX Buffer Bypass .....	97
TX Pattern Generator.....	105
TX Polarity Control .....	108
TX Fabric Clock Output Control .....	109
TX Phase Interpolator PPM Controller.....	113
TX Configurable Driver .....	116
TX Receiver Detect Support for PCI Express Designs .....	123
TX Out-of-Band Signaling.....	125

## Chapter 4: Receiver

RX Overview.....	127
RX Analog Front End.....	128
RX Out-of-Band Signaling .....	133
RX Equalizer.....	141
RX CDR .....	143
RX Fabric Clock Output Control.....	149
RX Margin Analysis.....	153
RX Polarity Control .....	161
RX Pattern Checker .....	162
RX Byte and Word Alignment.....	164
RX 8B/10B Decoder.....	173
RX Buffer Bypass .....	177
RX Elastic Buffer.....	190
RX Clock Correction .....	194
RX Channel Bonding.....	201
RX Gearbox.....	211
FPGA RX Interface.....	218

## Chapter 5: Board Design Guidelines

Overview .....	223
Pin Description and Design Guidelines .....	223
Reference Clock.....	230
Power Supply and Filtering .....	234
SelectIO Usage Guidelines.....	241
PCB Design Checklist.....	241

## Appendix A: Placement Information by Package

CPG236 Package Placement Diagram .....	244
CSG325 Package Placement Diagram .....	245
CLG485 Package Placement Diagram .....	246
FGG484 Package Placement Diagram .....	247
FGG676 Package Placement Diagram .....	248
FBG484 Package Placement Diagram .....	250
SBG484 Package Placement Diagram .....	251
FBG676 Package Placement Diagram .....	252
FFG1156 Package Placement Diagram .....	254

**Appendix B: Placement Information by Device**

**Appendix C: 8B/10B Valid Characters**

**Appendix D: DRP Address Map of the GTP Transceiver**



# About This Guide

---

Xilinx® 7 series FPGAs include three FPGA families that are all designed for lowest power to enable a common design to scale across families for optimal power, performance, and cost. The Artix™-7 family is optimized for lowest cost and absolute power for the highest volume applications. The Virtex®-7 family is optimized for highest system performance and capacity. The Kintex™-7 family is an innovative class of FPGAs optimized for the best price-performance. This guide serves as a technical reference describing the 7 series FPGAs GTP transceivers.

The 7 series FPGAs GTP transceivers user guide, part of an overall set of documentation on the 7 series FPGAs, is available on the Xilinx website at [www.xilinx.com/7](http://www.xilinx.com/7).

In this document:

- 7 series FPGAs GTP transceiver channel is abbreviated as GTP transceiver.
- GTPE2\_CHANNEL is the name of the instantiation primitive that instantiates one GTP transceiver channel.
- GTPE2\_COMMON is the name of the primitive that instantiates two ring oscillator PLLs (PLL0 and PLL1).
- A Quad or Q is a cluster or set of four GTP transceiver channels, one GTPE2\_COMMON primitive, two differential reference clock pin pairs, and analog supply pins.

## Guide Contents

This manual contains:

- [Chapter 1, Transceiver and Tool Overview](#)
- [Chapter 2, Shared Features](#)
- [Chapter 3, Transmitter](#)
- [Chapter 4, Receiver](#)
- [Chapter 5, Board Design Guidelines](#)
- [Appendix A, Placement Information by Package](#)
- [Appendix B, Placement Information by Device](#)
- [Appendix C, 8B/10B Valid Characters](#)
- [Appendix D, DRP Address Map of the GTP Transceiver](#)

## Additional Resources

To find additional documentation, see the Xilinx website at:

<http://www.xilinx.com/support/documentation/index.htm>.

To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:

<http://www.xilinx.com/support>.

## Additional References

These documents provide additional information useful to this document:

1. *High-Speed Serial I/O Made Simple*

<http://www.xilinx.com/publications/archives/books/serialio.pdf>

# *Transceiver and Tool Overview*

---

## Overview and Features

The 7 series FPGAs GTP transceiver is a power-efficient transceiver, supporting line rates between 500 Mb/s and 6.6 Gb/s. The GTP transceiver is highly configurable and tightly integrated with the programmable logic resources of the FPGA. [Table 1-1](#) summarizes the features by functional group that support a wide variety of applications.

*Table 1-1: 7 Series FPGAs Transceiver Features*

Group	Feature	GTP	GTX	GTH
PCS	2-byte internal datapath	x	x	x
	4-byte internal datapath		x	x
	8B/10B encoding and decoding	x	x	x
	64B/66B and 64B/67B support	x	x	x
	Comma detection and byte and word alignment	x	x	x
	PRBS generator and checker	x	x	x
	FIFO for clock correction and channel bonding	x	x	x
	Programmable FPGA logic interface	x	x	x
PMA	One shared LC tank PLL per Quad		x	x
	One ring oscillator PLL per channel		x	x
	Two shared ring oscillator PLLs per Quad	x		
	Flexible reference clocking options	x	x	x
	Decision feedback equalization (DFE)		x	x
	Power-efficient adaptive linear equalizer mode called the low-power mode (LPM)	x	x	x
	TX Pre-emphasis	x	x	x
	Beacon signaling for PCI Express® designs	x	x	x
	Out-of-band (OOB) signaling including COM signal support for Serial ATA (SATA) designs	x	x	x
	RX Margin Analysis	x	x	x

The GTP transceiver offers a data rate range and features that allow physical layer support for various protocols including:

- PCI Express, Revision 1.1/2.0
- Interlaken
- 10 Gb Attachment Unit Interface (XAUI), Reduced Pin eXtended Attachment Unit Interface (RXAUI)
- Common Packet Radio Interface (CPRI™)/Open Base Station Architecture Initiative (OBSAI)
- OC-48
- OTU-1
- Serial RapidIO (SRIO)
- Serial Advanced Technology Attachment (SATA)/Serial Attached SCSI (SAS)
- Serial Digital Interface (SDI)

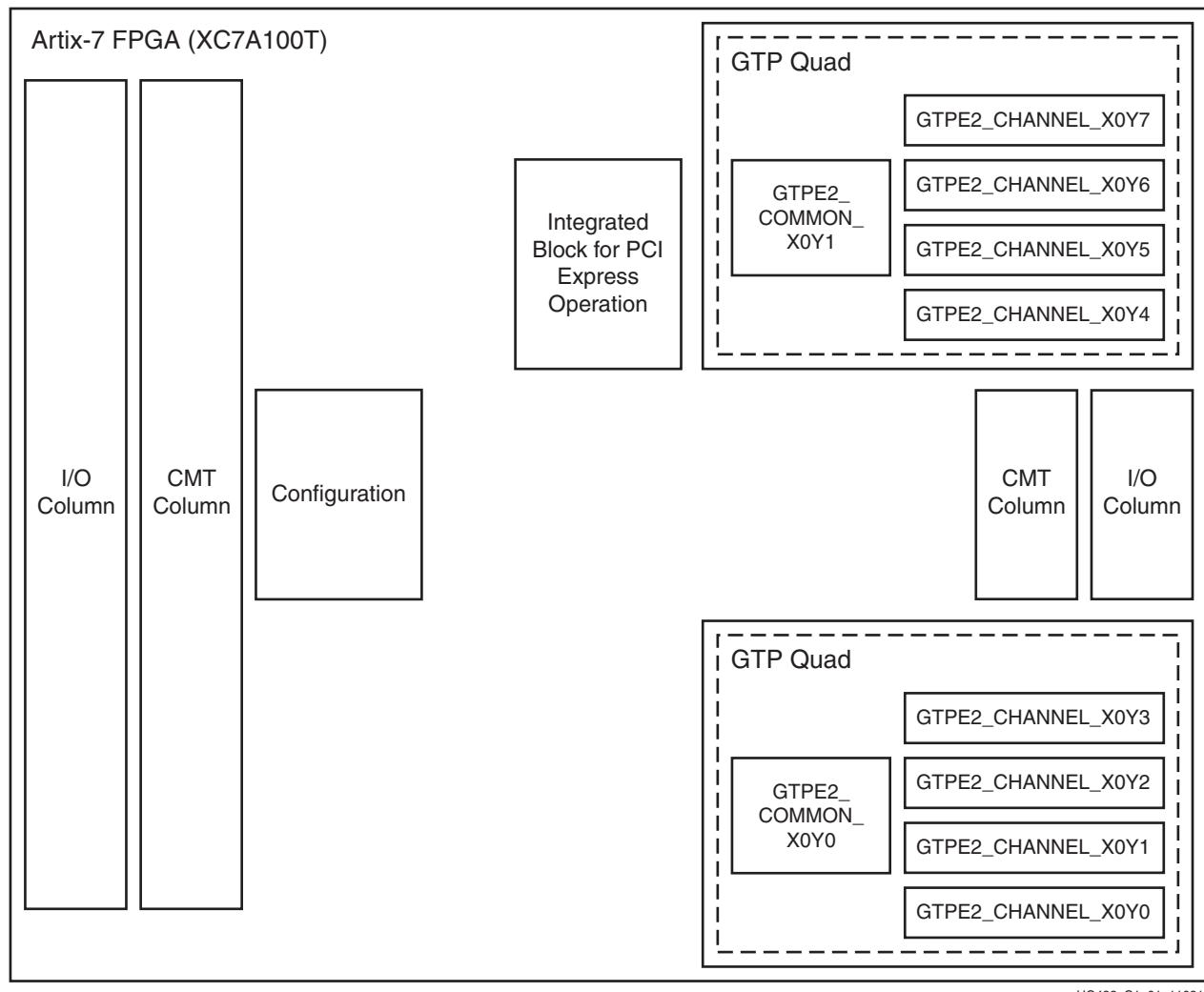
The CORE Generator™ tool includes a wizard to automatically generate predefined settings to configure GTP transceivers to support configurations for different protocols. The wizard can also be used to create custom configurations. For a complete list of protocols and electrical specifications enabled through predefined settings, please refer to [UG769, LogiCORE IP 7 Series FPGAs Transceivers Wizard User Guide](#).

In comparison to prior generation transceivers in Spartan®-6 FPGAs, the GTP transceiver in the 7 series FPGAs has the following new or enhanced features:

- 2-byte internal datapath
- Two ring oscillator PLLs per Quad
- Power-efficient, adaptive continuous time linear equalizer (CTLE)
- RX margin analysis feature to provide non-destructive, 2-D post-equalization eye scan.

The first-time user is recommended to read *High-Speed Serial I/O Made Simple* [Ref 1], which discusses high-speed serial transceiver technology and its applications.

[Figure 1-1, page 13](#) shows the GTP transceiver placement in an example Artix™-7 device (XC7A100T). This device has 8 GTP transceivers.



UG482\_C1\_01\_110811

**Figure 1-1: GTP Transceiver Inside Artix-7 XC7A100T FPGA**

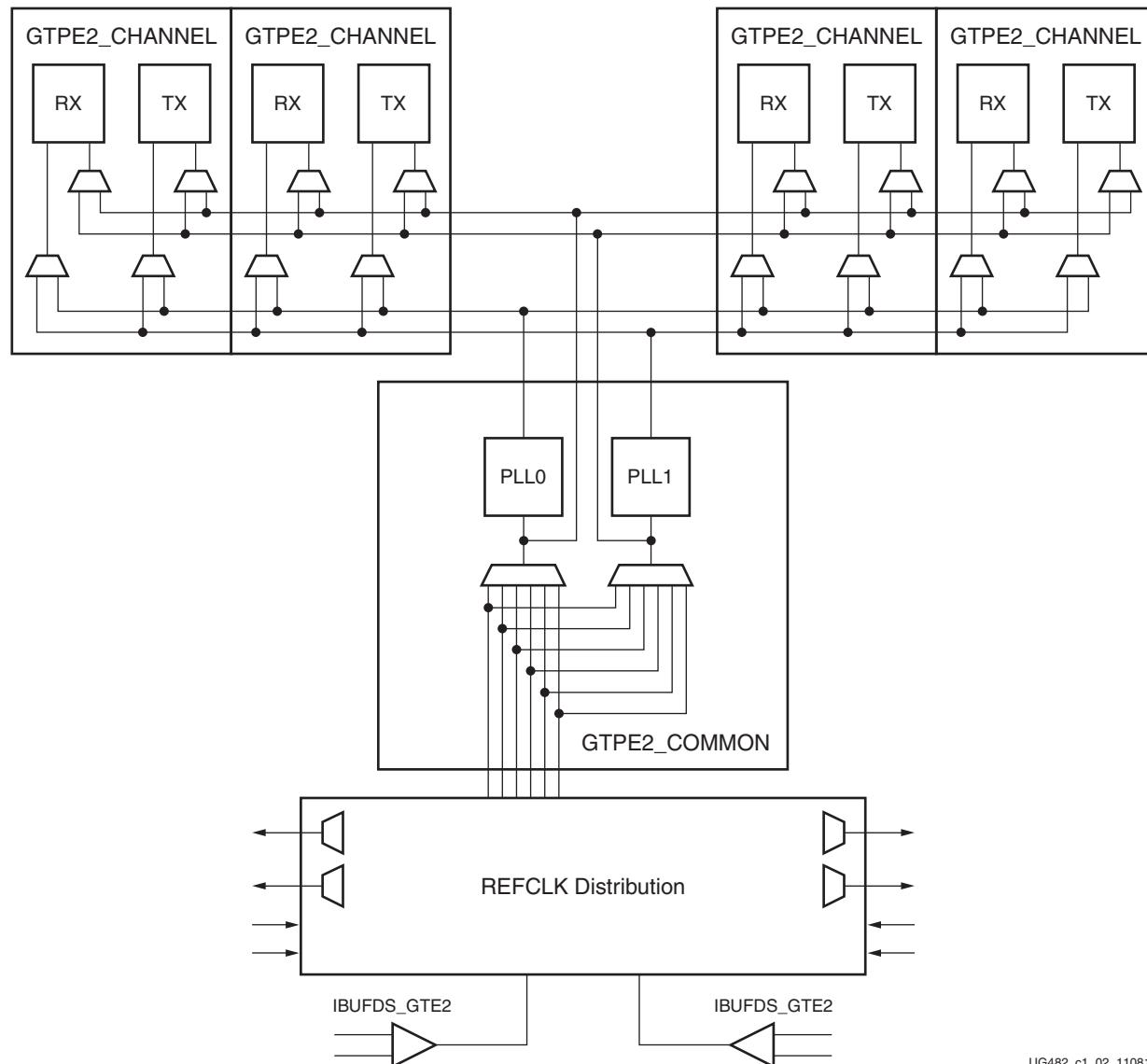
Additional information on the functional blocks of 7 series FPGAs is available at:

[UG470, 7 Series FPGAs Configuration User Guide](#) provides more information on the configuration.

[UG471, 7 Series FPGAs SelectIO Resources User Guide](#) provides more information on the I/O blocks.

[UG472, 7 Series FPGAs Clocking Resources User Guide](#) provides more information on the mixed mode clock manager (MMCM).

Figure 1-2 illustrates the clustering of four GTPE2\_CHANNEL primitives and one GTPE2\_COMMON primitive to form a Quad.



UG482\_c1\_02\_110811

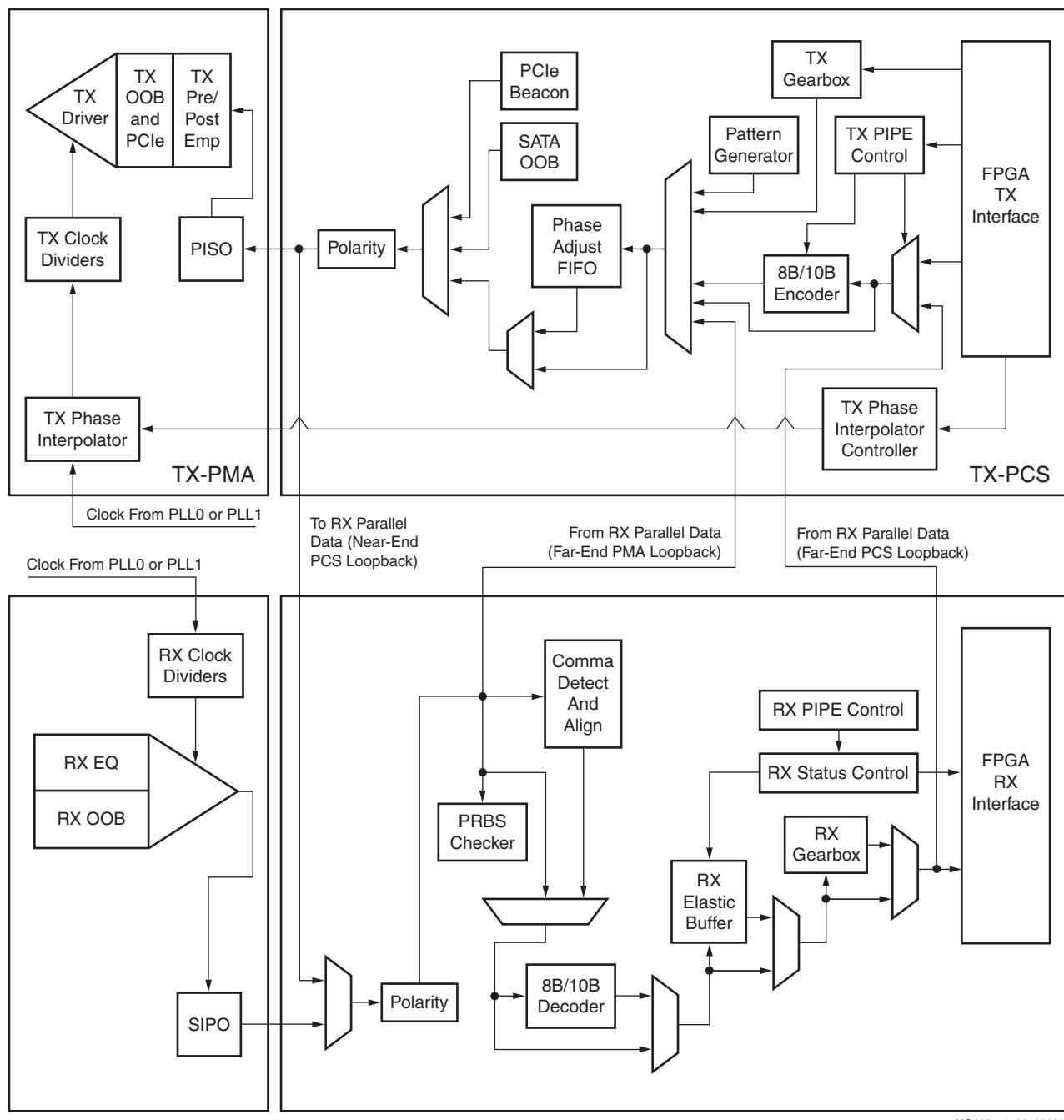
**Figure 1-2: GTP Transceiver Quad Configuration**

Four GTPE2 channels clustered together with one GTPE2\_COMMON primitive are called a *Quad* or *Q*.

The GTPE2\_COMMON primitive contains two ring oscillator PLLs (PLL0 and PLL1). GTPE2\_COMMON must always be instantiated.

Each GTPE2\_CHANNEL primitive consists of a transmitter and a receiver.

Figure 1-3 illustrates the topology of a GTPE2\_CHANNEL primitive.



UG482\_c1\_03\_110811

Figure 1-3: **GTPE2\_CHANNEL Primitive Topology**

Refer to [Figure 2-9, page 36](#) for the description of the channel clocking architecture, which provides clocks to the RX and TX clock dividers.

## 7 Series FPGAs Transceivers Wizard

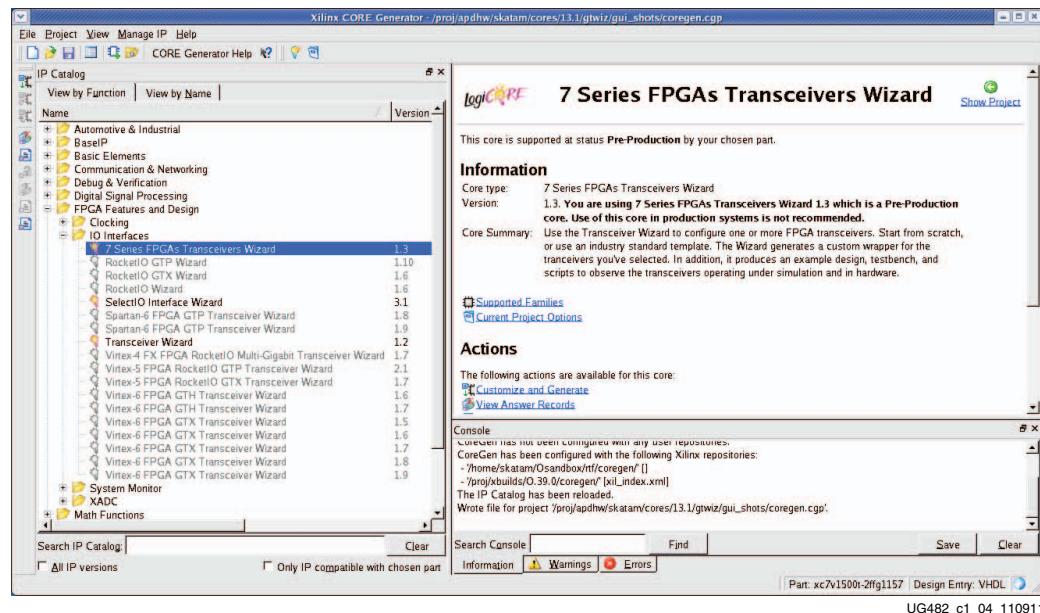
The 7 Series FPGAs Transceivers Wizard (hereinafter called the Wizard) is the preferred tool to generate a wrapper to instantiate GTP transceiver primitives called GTPE2\_COMMON and GTPE2\_CHANNEL. The Wizard is located in the CORE Generator tool. The user is recommended to download the most up-to-date IP update before using the Wizard. Details on how to use this Wizard can be found in [UG769, LogiCORE IP 7 Series FPGAs Transceivers Wizard User Guide](#).

Follow these steps to launch the Wizard:

1. Start the CORE Generator tool.
2. Locate the 7 Series FPGAs Transceivers Wizard in the taxonomy tree under:

/FPGA Features & Design/IO Interfaces

See [Figure 1-4](#).



**Figure 1-4: 7 Series FPGAs Transceivers Wizard**

3. Double-click 7 Series FPGAs Transceivers Wizard to launch the Wizard.

## Simulation

### Functional Description

Simulations using the GTPE2\_CHANNEL and GTPE2\_COMMON primitives have specific prerequisites that the simulation environment and the test bench must fulfill. For instructions on how to set up the simulation environment for supported simulators depending on the used hardware description language (HDL), see the latest version of [UG626, Synthesis and Simulation Design Guide](#).

The prerequisites for simulating a design with the GTPE2\_CHANNEL and GTPE2\_COMMON primitives are:

- A simulator with support for SecureIP models.  
SecureIP models are encrypted versions of the Verilog HDL used for implementation of the modeled block. SecureIP is an IP encryption methodology. To support SecureIP models, a Verilog LRM - IEEE Std 1364-2005 encryption compliant simulator is required.
- A mixed-language simulator for VHDL simulation.  
SecureIP models use a Verilog standard. To use them in a VHDL design, a mixed-language simulator is required. The simulator must be able to simulate VHDL and Verilog simultaneously.
- An installed GTP transceiver SecureIP model.
- The correct setup of the simulator for SecureIP use (initialization file, environment variables).
- The ability to run COMPXLIB, which compiles the simulation libraries (e.g., UNISIM, SIMPRIMS) in the correct order.
- The correct simulator resolution (Verilog).
- The user guide of the simulator and [UG626, Synthesis and Simulation Design Guide](#) provide a detailed list of settings for SecureIP support.

## Ports and Attributes

There are no simulation-only ports on the GTPE2\_COMMON and GTPE2\_CHANNEL primitives.

### GTPE2\_COMMON Attributes

The GTPE2\_COMMON primitive has attributes intended only for simulation. [Table 1-2](#) lists the simulation-only attributes of the GTPE2\_COMMON primitive. The names of these attributes start with *SIM\_*.

**Table 1-2: GTPE2\_COMMON Simulation-Only Attributes**

Attribute	Type	Description
SIM_PLL0REFCLK_SEL	3-bit Binary	This attribute selects the reference clock source used to drive PLL0 in simulation for designs where PLL0 is always driven by the same reference clock source. SIM_PLL0REFCLK_SEL allows for simulation before and after the port swap changes. This allows for the block to be simulated with the correct clock source both before and after the port swap. SIM_PLL0REFCLK_SEL must be set to the same value as PLL0REFCLK SEL[2:0]. For designs that require the reference clock source to be changed on the fly, the port PLL0REFCLKSEL is used instead to dynamically select the reference clock source.
SIM_PLL1REFCLK_SEL	3-bit Binary	This attribute selects the reference clock source used to drive PLL1 in simulation for designs where PLL1 is always driven by the same reference clock source. SIM_PLL1REFCLK_SEL allows for simulation before and after the port swap changes. This allows for the block to be simulated with the correct clock source both before and after the port swap. SIM_PLL1REFCLK_SEL must be set to the same value as PLL1REFCLK SEL[2:0]. For designs that require the reference clock source to be changed on the fly, the port PLL1REFCLKSEL is used instead to dynamically select the reference clock source.

**Table 1-2: GTPE2\_COMMON Simulation-Only Attributes (Cont'd)**

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
SIM_RESET_SPEEDUP	String	If the SIM_RESET_SPEEDUP attribute is set to TRUE (default), an approximated reset sequence is used to speed up the reset time for simulations, where faster reset times and faster simulation times are desirable. If the SIM_RESET_SPEEDUP attribute is set to FALSE, the model emulates hardware reset behavior in detail.
SIM_VERSION	String	This attribute selects the simulation version to match different steppings of silicon. The default for this attribute is 1.0.

## GTPE2\_CHANNEL Attributes

The GTPE2\_CHANNEL primitive has attributes intended only for simulation. [Table 1-3](#) lists the simulation-only attributes of the GTPE2\_CHANNEL primitive. The names of these attributes start with SIM\_.

**Table 1-3: GTPE2\_CHANNEL Simulation-Only Attributes**

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
SIM_RESET_SPEEDUP	String	If the SIM_RESET_SPEEDUP attribute is set to TRUE (default), an approximated reset sequence is used to speed up the reset time for simulations, where faster reset times and faster simulation times are desirable. If the SIM_RESET_SPEEDUP attribute is set to FALSE, the model emulates hardware reset behavior in detail.
SIM_RECEIVER_DETECT_PASS	String	SIM_RECEIVER_DETECT_PASS is a string TRUE/FALSE attribute to determine if a receiver detect operation should indicate a pass or fail in simulation.
SIM_TX_EIDLE_DRIVE_LEVEL	String	SIM_TX_EIDLE_DRIVE_LEVEL can be set to 0, 1, X, or Z to allow for simulation of electrical idle and receiver detect operations using an external pull-up resistor. The default for this attribute is X.
SIM_VERSION	String	This attribute selects the simulation version to match different steppings of silicon. The default for this attribute is 1.0.

# Implementation

## Functional Description

This section provides the information needed to map 7 series GTP transceivers instantiated in a design to device resources, including:

- The location of the GTP transceiver Quads on the available device and package combinations.
- The pad numbers of external signals associated with each GTP transceiver Quad.
- How the GTPE2\_CHANNEL primitive, the GTPE2\_COMMON primitive, and clocking resources instantiated in a design are mapped to available locations with a user constraints file (UCF).

It is a common practice to define the location of GTP transceiver Quads early in the design process to ensure correct usage of clock resources and to facilitate signal integrity analysis during board design. The implementation flow facilitates this practice through the use of location constraints in the UCF.

This section describes how to instantiate GTP transceiver clocking components.

The position of each GTP transceiver channel and common primitive is specified by an XY coordinate system that describes the column number and the relative position within that column.

For a given device/package combination, the transceiver with the coordinates X0Y0 is always located at the lowest position of the lowest available bank.

There are two ways to create a UCF for designs that utilize the GTP transceiver. The preferred method is to use the 7 Series FPGAs Transceivers Wizard. The Wizard automatically generates UCF templates that configure the transceivers and contain placeholders for GTP transceiver placement information. The UCFs generated by the Wizard can then be edited to customize operating parameters and placement information for the application.

The second approach is to create the UCF by hand. When using this approach, the designer must enter both configuration attributes that control transceiver operation as well as tile location parameters. Care must be taken to ensure that all of the parameters needed to configure the GTP transceiver are correctly entered.

If a design requires the use of any of the GTP channels in a given GTP Quad, a GTPE2\_COMMON primitive must be instantiated as shown in [Figure 1-5](#). At a minimum, at least one GTPE2\_CHANNEL must also be instantiated. [Figure 1-5](#) shows four GTPE2\_CHANNEL primitives instantiated.

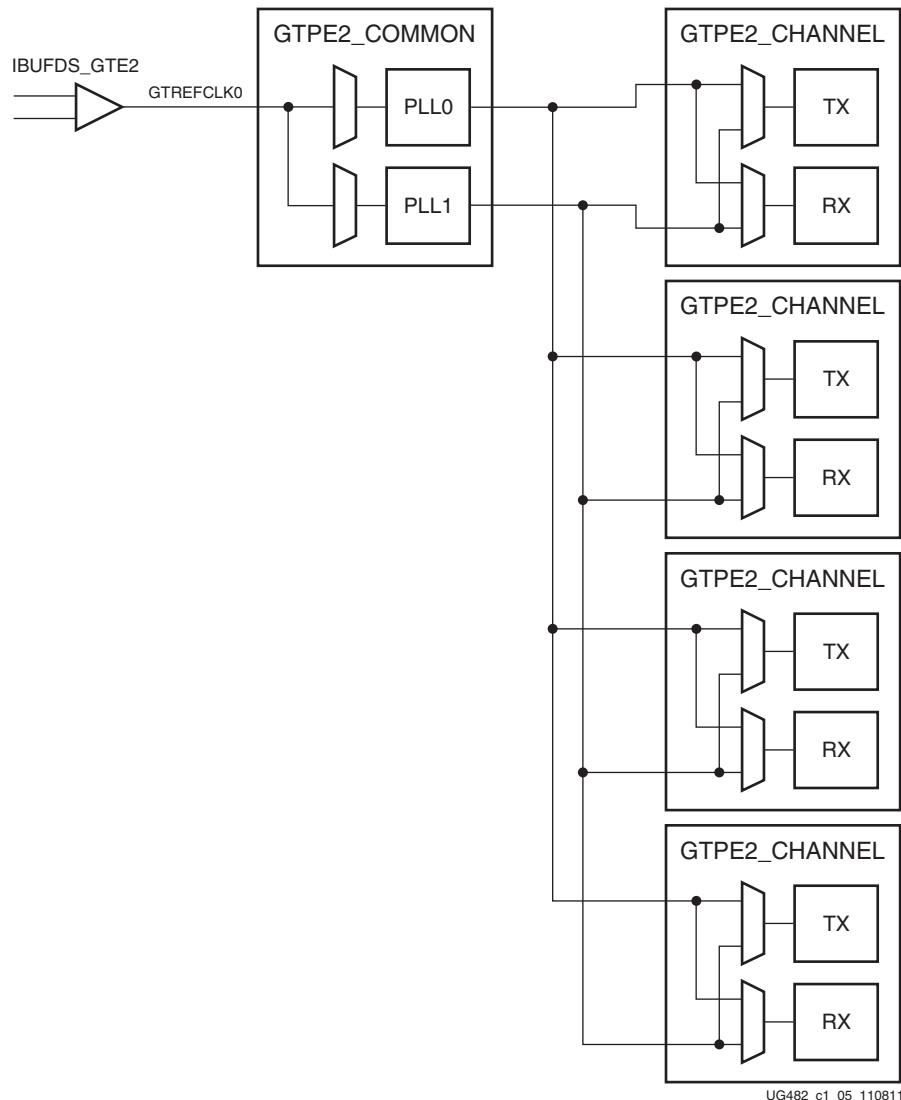


Figure 1-5: Four Channel Configuration

## Serial Transceiver Channels by Device/Package

See [UG475, 7 Series FPGAs Packaging and Pinout Specification](#).



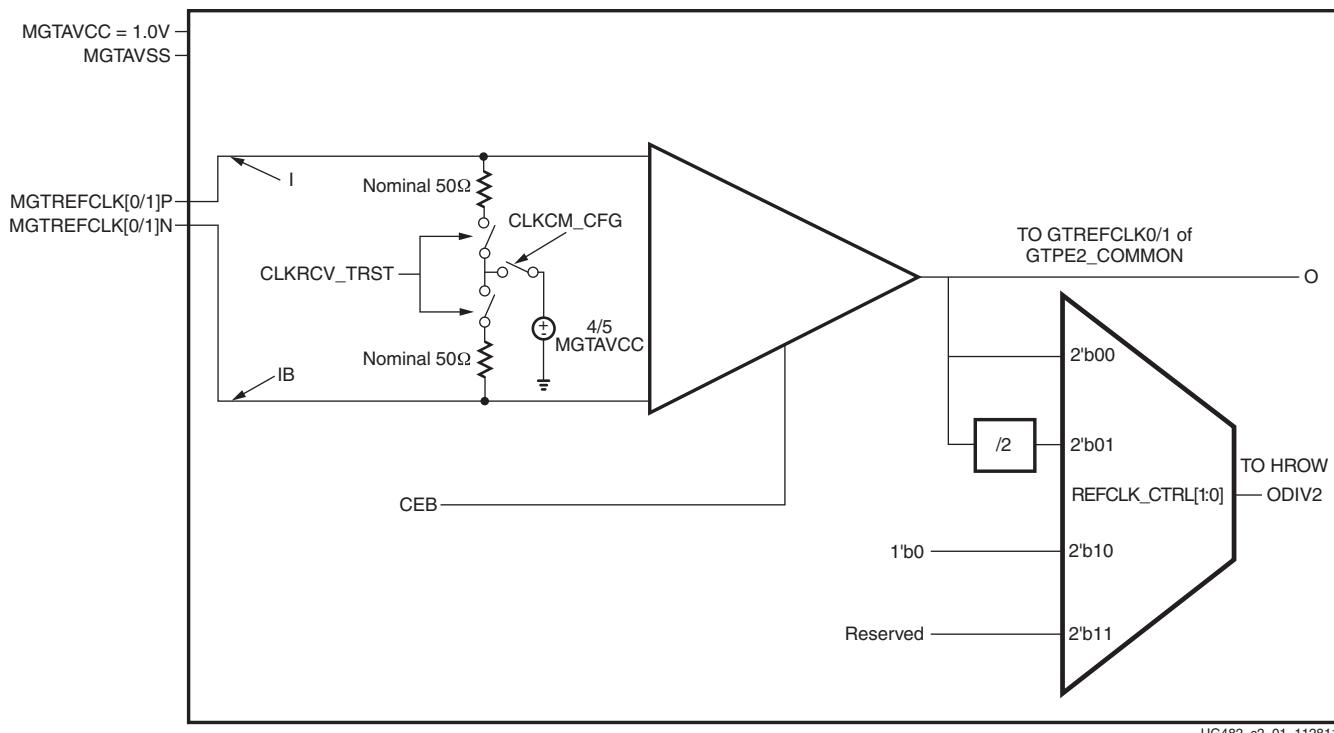
# Shared Features

## Reference Clock Input Structure

### Functional Description

The reference clock input structure is illustrated in [Figure 2-1](#). The input is terminated internally with  $50\Omega$  on each leg to 4/5 MGTAVCC. The reference clock is instantiated in software with the IBUFDS\_GTE2 software primitive. The ports and attributes controlling the reference clock input are tied to the IBUFDS\_GTE2 software primitive.

[Figure 2-1](#) shows the internal structure of the reference clock input buffer.



*Figure 2-1: Reference Clock Input Structure*

## Ports and Attributes

**Table 2-1** defines the reference clock input ports in the IBUFDS\_GTE2 software primitive.

**Table 2-1: Reference Clock Input Ports (IBUFDS\_GTE2)**

Port	Dir	Clock Domain	Description
I IB	In (pad)	N/A	These are the reference clock input ports that get mapped to GTREFCLK0P/GTREFCLK0N and GTREFCLK1P/GTREFCLK1N.
CEB	In	N/A	This is the active-Low asynchronous clock enable signal for the clock buffer. Setting this signal High powers down the clock buffer.
O	Out	N/A	<p>This output drives the GTREFCLK[0/1] signals in the GTPE2_COMMON software primitives. Refer to <a href="#">Reference Clock Selection and Distribution, page 26</a> for more details.</p> <p>This output can also drive the BUFG or BUFH software primitives via Hrow routing. Only one of the IBUFDS_GTE2's O or ODIV2 outputs can be routed to the FPGA logic. The selection is controlled automatically by the software depending on whether port O or ODIV2 is connected. Refer to <a href="#">UG472, 7 Series FPGAs Clocking Resources User Guide</a> for more details.</p>
ODIV2 <sup>(1)</sup>	Out	N/A	<p>This output is a divide-by-2 version of the O signal, which can drive the BUFG or BUFH software primitives via Hrow routing. The selection is controlled automatically by the software depending on whether port O or ODIV2 is connected. Refer to <a href="#">UG472, 7 Series FPGAs Clocking Resources User Guide</a> for more details.</p>

**Notes:**

1. The O and ODIV2 outputs are not phase matched to each other.

**Table 2-2** defines the attributes in the IBUFDS\_GTE2 software primitive that configure the reference clock input.

**Table 2-2: Reference Clock Input Attributes (IBUFDS\_GTE2)**

Attribute	Type	Description
CLK_RCV_TRST	Boolean	Reserved. This attribute switches the 50Ω termination resistors into the signal path. This attribute must always be set to TRUE.

**Table 2-2: Reference Clock Input Attributes (IBUFDS\_GTE2) (Cont'd)**

Attribute	Type	Description
CLKCM_CFG	Boolean	Reserved. This attribute switches in the termination voltage for the 50Ω termination. This attribute must always be set to TRUE.
CLKSWING_CFG	2-bit Binary	Reserved. This attribute controls the internal swing of the clock. This attribute must always be set to 2'b11.

## Use Modes: Reference Clock Termination

The reference clock input is to be externally AC coupled. [Table 2-3](#) shows the port and attribute settings required to achieve this.

**Table 2-3: Port and Attribute Settings**

Input Type	Settings
Ports	CEB = 0
Attributes	CLKRCV_TRST = TRUE CLKCM_CFG = TRUE CLKSWING_CFG = 2'b11

## Reference Clock Selection and Distribution

### Functional Description

The GTP transceivers in 7 series FPGAs provide different reference clock input options. Clock selection and availability differs slightly from 7 series GTX/GTH transceivers in that reference clock routing is east and west bound rather than north and south bound.

Architecturally, the concept of a Quad (or Q), contains a grouping of four GTPE2\_CHANNEL primitives, one GTPE2\_COMMON primitive, two dedicated external reference clock pin pairs, and dedicated reference clock routing. The GTPE2\_COMMON primitive must always be instantiated, and the GTPE2\_CHANNEL primitive must be instantiated for each transceiver. For the largest Artix™-7 device (XC7A200T-FFG1156), the reference clock supplied to the PLLs in a given Quad can also be sourced from the adjacent Quad in the same half of the device. A Quad located in the top half of the device can share its two local reference clocks with the other Quad located in the top half. Similarly, a Quad located in the bottom half of the device can share its two reference clocks with the other Quad located in the bottom half.

Reference clock features include:

- Clock routing for east and west bound clocks.
- Flexible reference clock inputs available for PLL0 and PLL1.
- Static or dynamic selection of the reference clock for PLL0 and PLL1.

[Figure 2-2](#) shows the reference clock architecture with the GTPE2\_COMMON primitive, two dedicated reference clock pin pairs, and dedicated east or west reference clock routing. Each GTPE2\_COMMON in a Quad has four clock inputs available:

- Two local reference clock pin pairs, GTREFCLK0 or GTREFCLK1
- Two reference clock pin pairs from the other Quad situated in the same half of the device

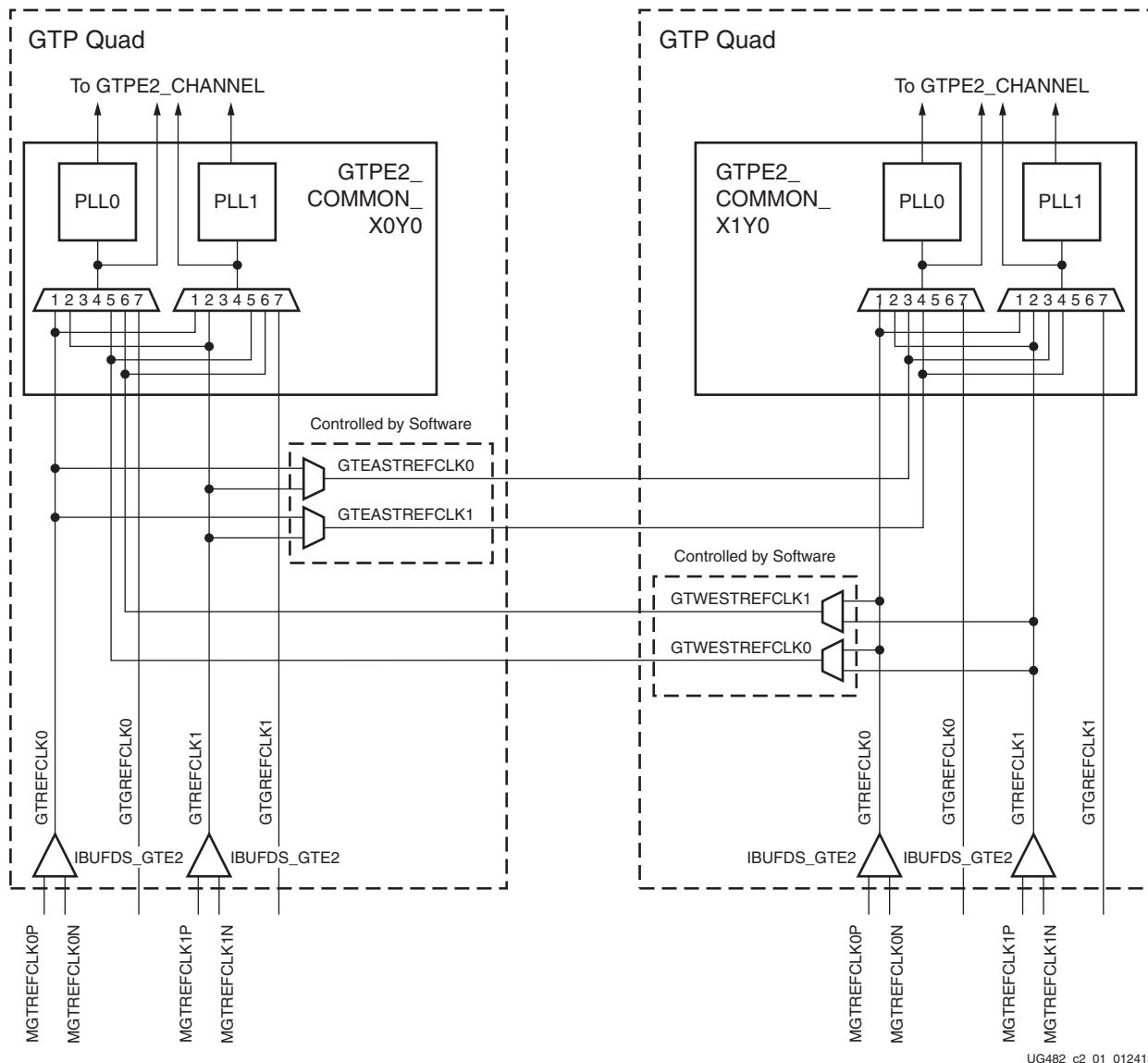
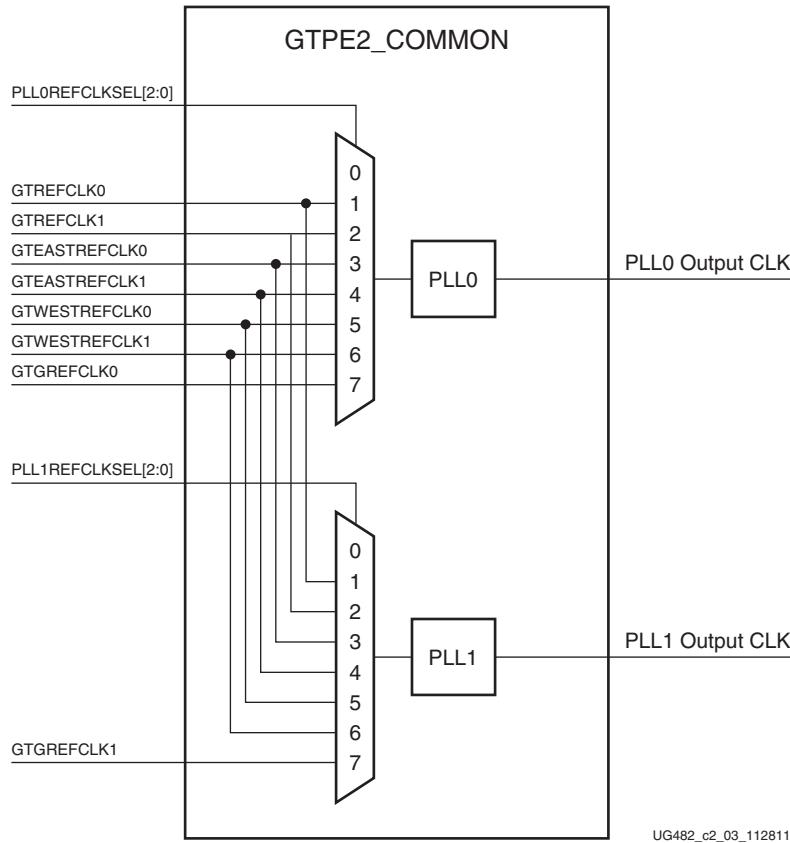


Figure 2-2: Conceptual View of GTP Transceiver Reference Clocking

Figure 2-3 shows the detailed view of the reference clock multiplexer structures within a single GTPE2\_COMMON primitive. The PLL0REFCLKSEL and PLL1REFCLKSEL ports are required when multiple reference clock sources are connected to the multiplexers. A single reference clock is most commonly used. In this case, the PLL[0/1]REFCLKSEL port can be tied to 3'b001, and the Xilinx software tools handle the complexity of the multiplexers and associated routing. See [External Reference Clock Use Model, page 32](#) for more information.

UG482\_c2\_01\_012413



UG482\_c2\_03\_112811

Figure 2-3: **PLL0 and PLL1 Reference Clock Selection Multiplexer**

## Ports and Attributes

[Table 2-4](#) and [Table 2-5](#) define the clocking ports and attributes for the GTPE2\_COMMON primitive.

Table 2-4: **GTPE2\_COMMON Clocking Ports**

Port	Direction	Clock Domain	Description
GTGREFCLK0	In	Clock	Reference clock generated by the internal FPGA logic. This input is reserved for internal testing purposes only.
GTGREFCLK1	In	Clock	Reference clock generated by the internal FPGA logic. This input is reserved for internal testing purposes only.
GTREFCLK0	In	Clock	External clock driven by IBUFDS_GTE2 for PLL0 and/or PLL1.
GTREFCLK1	In	Clock	External clock driven by IBUFDS_GTE2 for PLL0 and/or PLL1.
GTWESTREFCLK0	In	Clock	West-bound clock from the Quad on the right side of the device.

Table 2-4: GTPE2\_COMMON Clocking Ports (*Cont'd*)

Port	Direction	Clock Domain	Description
GTWESTREFCLK1	In	Clock	West-bound clock from the Quad on the right side of the device.
GTEASTREFCLK0	In	Clock	East-bound clock from the Quad on the left side of the device.
GTEASTREFCLK1	In	Clock	East-bound clock from the Quad on the left side of the device.
PLL0OUTCLK	Out	Clock	PLL0 clock output. The user must connect this port to the PLL0CLK port on the GTPE2_CHANNEL primitive.
PLL1OUTCLK	Out	Clock	PLL1 clock output. The user must connect this port to the PLL1CLK port on the GTPE2_CHANNEL primitive.
PLL0OUTREFCLK	Out	Clock	The user must connect this port to the PLL0REFCLK port on the GTPE2_CHANNEL primitive.
PLL1OUTREFCLK	Out	Clock	The user must connect this port to the PLL1REFCLK port on the GTPE2_CHANNEL primitive.

Table 2-4: GTPE2\_COMMON Clocking Ports (Cont'd)

Port	Direction	Clock Domain	Description
PLL0REFCLKSEL[2:0]	In	Async	<p>Input to dynamically select the input reference clock to PLL0. This input should be set to 3'b001 when only one clock source is connected to the PLL0 reference clock selection multiplexer.</p> <p>Reset must be applied to PLL0 after changing the reference clock input.</p> <ul style="list-style-type: none"> <li>000: Reserved</li> <li>001: GTREFCLK0 selected</li> <li>010: GTREFCLK1 selected</li> <li>011: GTEASTREFCLK0 selected</li> <li>100: GTEASTREFCLK1 selected</li> <li>101: GTWESTREFCLK0 selected</li> <li>110: GTWESTREFCLK1 selected</li> <li>111: GTGREFCLK0 selected</li> </ul>
PLL1REFCLKSEL[2:0]	In	Async	<p>Input to dynamically select the input reference clock to PLL1. This input should be set to 3'b001 when only one clock source is connected to the PLL1 reference clock selection multiplexer.</p> <p>Reset must be applied to PLL1 after changing the reference clock input.</p> <ul style="list-style-type: none"> <li>000: Reserved</li> <li>001: GTREFCLK0 selected</li> <li>010: GTREFCLK1 selected</li> <li>011: GTEASTREFCLK0 selected</li> <li>100: GTEASTREFCLK1 selected</li> <li>101: GTWESTREFCLK0 selected</li> <li>110: GTWESTREFCLK1 selected</li> <li>111: GTGREFCLK1 selected</li> </ul>

Table 2-5: GTPE2\_COMMON Attributes

Attribute	Type	Description
SIM_PLL0REFCLK_SEL	3-bit Binary	This attribute selects the reference clock source used to drive PLL0 in simulation for designs where PLL0 is always driven by the same reference clock source. SIM_PLL0REFCLK_SEL allows for simulation before and after the port swap changes. This allows for the block to be simulated with the correct clock source both before and after the port swap. SIM_PLL0REFCLK_SEL must be set to the same value as PLL0REFCLK SEL[2:0]. For designs that require the reference clock source to be changed on the fly, the port PLL0REFCLKSEL is used instead to dynamically select the reference clock source.
SIM_PLL1REFCLK_SEL	3-bit Binary	This attribute selects the reference clock source used to drive PLL1 in simulation for designs where PLL1 is always driven by the same reference clock source. SIM_PLL1REFCLK_SEL allows for simulation before and after the port swap changes. This allows for the block to be simulated with the correct clock source both before and after the port swap. SIM_PLL1REFCLK_SEL must be set to the same value as PLL1REFCLK SEL[2:0]. For designs that require the reference clock source to be changed on the fly, the port PLL1REFCLKSEL is used instead to dynamically select the reference clock source.

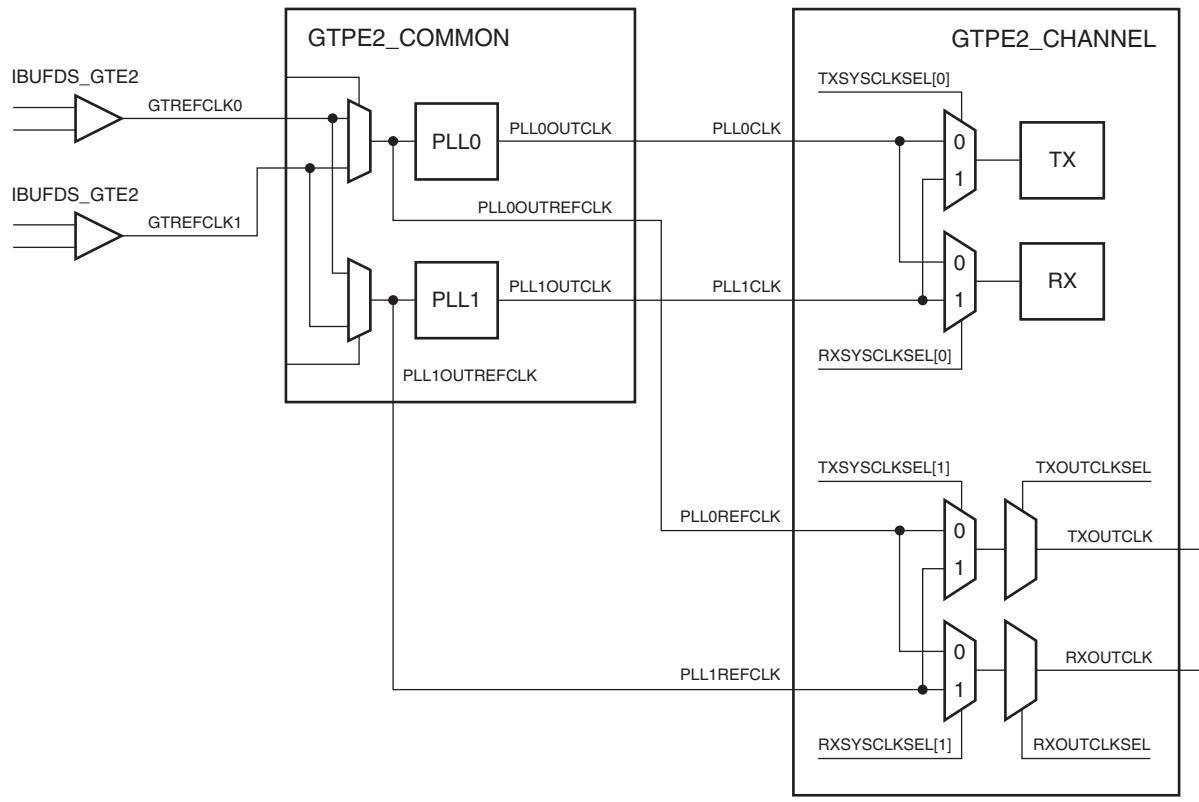
Table 2-6 defines the clocking ports for the GTPE2\_CHANNEL primitive.

Table 2-6: GTPE2\_CHANNEL Clocking Ports

Port	Direction	Clock Domain	Description
RXSYSCLKSEL[1:0]	In	Async	Selects the PLL clock source to drive the RX datapath: RXSYSCLKSEL[0] = 1'b0 (PLL0) RXSYSCLKSEL[0] = 1'b1 (PLL1) Selects the reference clock source to drive RXOUTCLK: RXSYSCLKSEL[1] = 1'b0 (reference clock from PLL0) RXSYSCLKSEL[1] = 1'b1 (reference clock from PLL1)
TXSYSCLKSEL[1:0]	In	Async	Selects the PLL clock source to drive the TX datapath: TXSYSCLKSEL[0] = 1'b0 (PLL0) TXSYSCLKSEL[0] = 1'b1 (PLL1) Selects the reference clock source to drive TXOUTCLK: TXSYSCLKSEL[1] = 1'b0 (reference clock from PLL0) TXSYSCLKSEL[1] = 1'b1 (reference clock from PLL1)
PLL0CLK	In	Clock	PLL0 clock input. The user must connect this port to the PLL0OUTCLK port on the GTPE2_COMMON primitive.
PLL1CLK	In	Clock	PLL1 clock input. The user must connect this port to the PLL1OUTCLK port on the GTPE2_COMMON primitive.
PLL0REFCLK	In	Clock	The user must connect this port to the PLL0OUTREFCLK port on the GTPE2_COMMON primitive.
PLL1REFCLK	In	Clock	The user must connect this port to the PLL1OUTREFCLK port on the GTPE2_COMMON primitive.

## External Reference Clock Use Model

Each Quad has two dedicated differential reference clock inputs that can be connected to external reference clock sources. An IBUFDS\_GTE2 primitive must be instantiated to use these dedicated reference clock pin pairs. The user design connects the IBUFDS\_GTE2 output (O) to the GTREFCLK[0/1], GTEASTREFCLK[0/1] or GTWESTREFCLK[0/1] ports of the GTPE2\_COMMON primitive where the reference clock selection multiplexer is located. Depending on the line rate requirement, the user design has the flexibility to use different combinations of PLL0 or PLL1 to drive the TX and/or RX datapath, as shown in Figure 2-4.

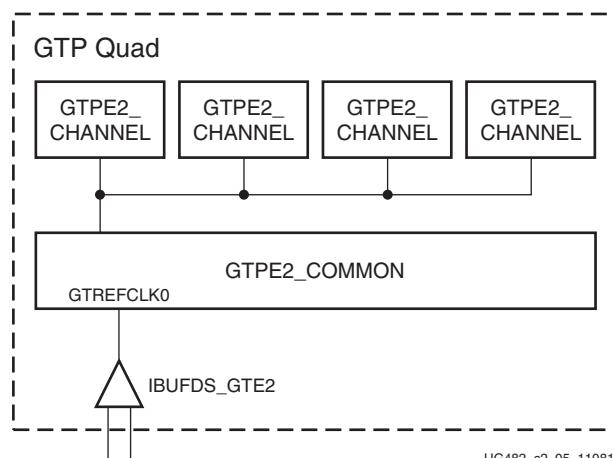


UG482\_c2\_04\_110811

Figure 2-4: External Reference Clock Use Case

## Single External Reference Clock Use Model

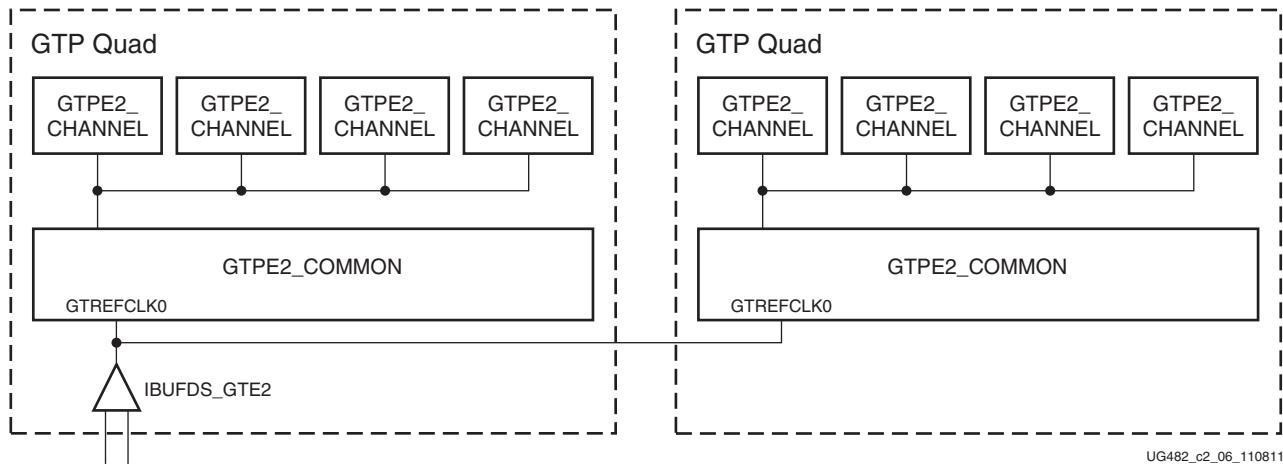
In the single external reference clock use model, the user connects the IBUFDs\_GTE2 output (O) to the GTREFCLK0 input port of the GTPE2\_COMMON primitive. The user design can leave the other unused reference clock ports floating. The IBUFDs\_GTE2 input pins can be constrained in the user constraints file (UCF). [Figure 2-5](#) shows a single GTPE2\_COMMON primitive connected to a single IBUFDs\_GTE2 primitive.



UG482\_c2\_05\_110811

Figure 2-5: Single GTP Quad with a Single Local Reference Clock

[Figure 2-6](#) shows a single reference clock connected to two GTP Quads. The user connects the IBUFDS\_GTE2 output (O) to the GTREFCLK0 input port of both GTPE2\_COMMON primitive instances. Such a scenario is only possible in the largest Artix-7 device (XC7A200T-FFG1156) that contains east and west GTP Quads adjacent to each other.

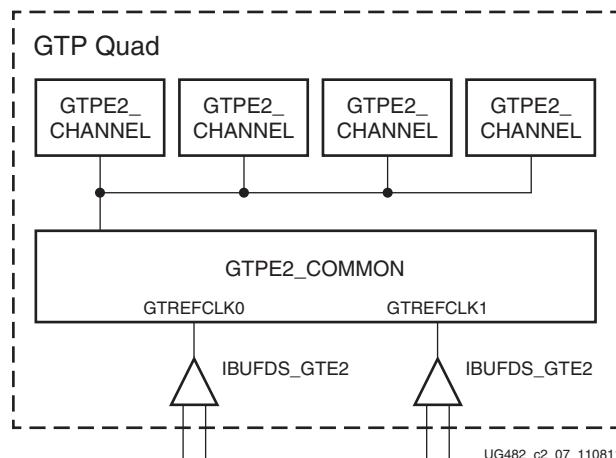


**Figure 2-6: Two GTP Quads with a Single Shared Reference Clock**

When required, as is the case for the design in [Figure 2-6](#), the Xilinx implementation tools make the necessary adjustments to the east/west routing shown in [Figure 2-2, page 27](#), as well as any necessary pin swapping to the GTPE2\_COMMON clock inputs to route the reference clocks between two Quads.

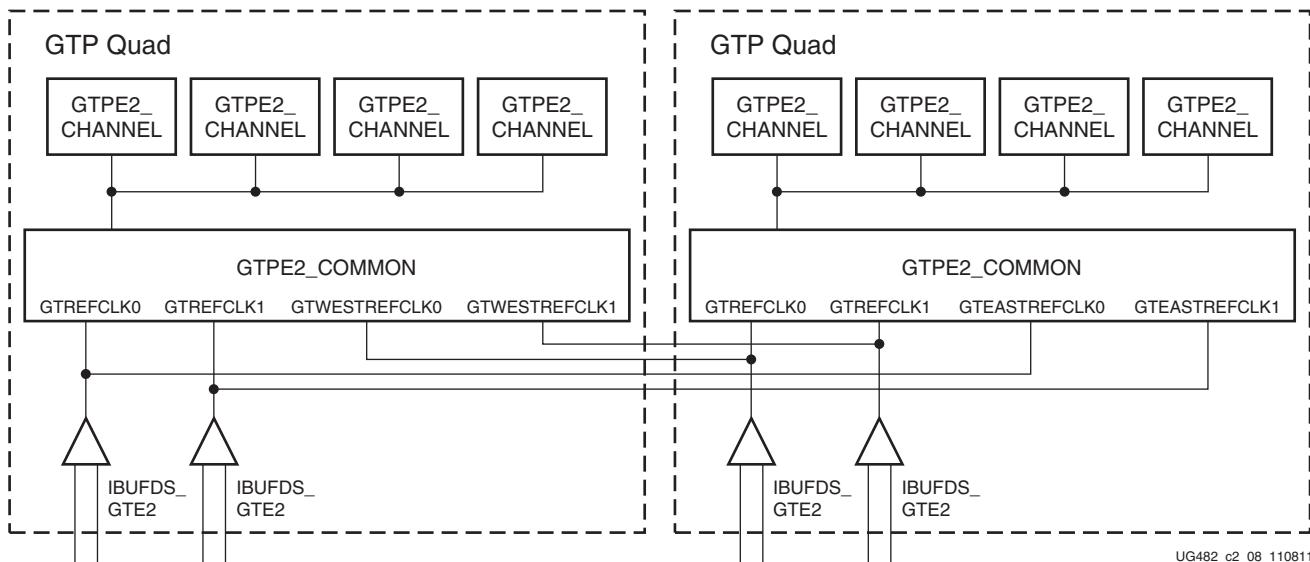
### Multiple External Reference Clock Use Model

In [Figure 2-7](#) and [Figure 2-9](#), because the reference clock multiplexer structures in the GTPE2\_COMMON have more than one reference clock source, the user design is required to connect the output of the IBUFDS\_GTE2 to the correct clock input ports on the GTPE2\_COMMON primitive. [Figure 2-7](#) shows an example of a single GTP Quad using both of its dedicated differential reference clock inputs. Two IBUFDS\_GTE2 primitives and a single GTPE2\_COMMON primitive are instantiated.



**Figure 2-7: Single GTP Quad using Multiple Local Reference Clocks**

**Figure 2-8** shows two GTP Quads, each utilizing their own dedicated differential reference clock inputs as well as the dedicated differential reference clock inputs of their neighboring GTP Quad. Such a scenario is only possible in the largest Artix-7 device (XC7A200T-FFG1156) that contains east and west GTP Quads adjacent to each other. The user is responsible for properly connecting the output of the IBUFDS\_GTE2 to the appropriate GTREFCLK[0/1], GTWESTREFCLK[0/1], and GTEASTREFCLK[0/1] input ports on the GTPE2\_COMMON primitive.



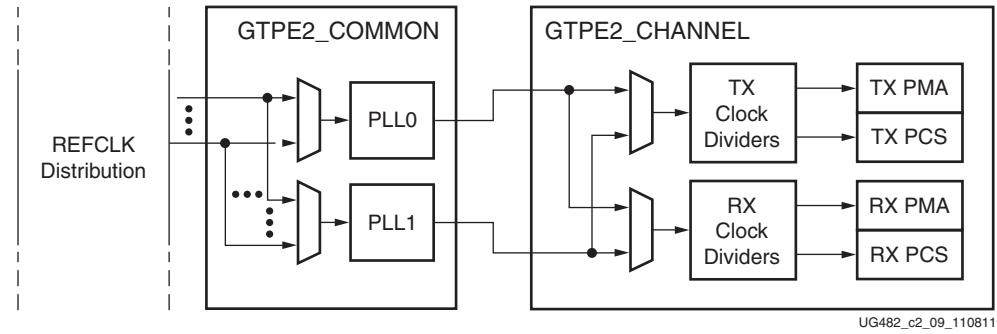
**Figure 2-8: Two GTP Quads using Multiple Reference Clocks from Different Quads**

For multi-rate designs that require the reference clock to be changed in real time, the PLL0REFCLKSEL and PLL1REFCLKSEL ports are used to dynamically select the reference clock source. When the selection has been made, the user design is responsible for resetting the PLL via PLL0RESET or PLL1RESET.

## PLL

### Functional Description

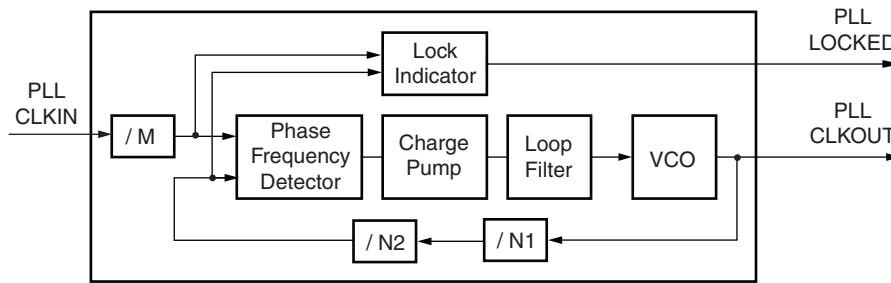
The GTP Quad contains two ring oscillator PLLs (PLL0 and PLL1). The internal clocking architecture is shown in **Figure 2-9**. When the TX and RX datapaths operate in the same line rate range, PLL0 or PLL1 can be shared between the TX and RX datapaths. The TX and RX clock dividers can individually select the clock from PLL0 or PLL1 to allow the TX and RX datapaths to operate at asynchronous frequencies using different reference clock inputs.



**Figure 2-9: Internal Clocking Architecture**

The PLL input clock selection is described in [Reference Clock Selection and Distribution, page 26](#). The PLL outputs feed the TX and RX clock divider blocks, which control the generation of serial and parallel clocks used by the PMA and PCS blocks.

[Figure 2-10](#) illustrates a conceptual view of the PLL architecture. The input clock can be divided by a factor of M before feeding into the phase frequency detector. The feedback dividers N1 and N2 determine the VCO multiplication ratio and the PLL output frequency. A lock indicator block compares the frequencies of the reference clock and the VCO feedback clock to determine if a frequency lock has been achieved.



**Figure 2-10: PLL Block Diagram**

The PLL has a nominal operating range between 1.6 GHz to 3.3 GHz. The 7 Series FPGAs Transceivers Wizard chooses the appropriate PLL settings based on application requirements.

[Equation 2-1](#) shows how to determine the PLL output frequency (GHz).

$$f_{PLLClkout} = f_{PLLClkin} \times \frac{N1 \times N2}{M} \quad \text{Equation 2-1}$$

[Equation 2-2](#) shows how to determine the line rate (Gb/s). D represents the value of the TX or RX clock divider block in the channel.

$$f_{LineRate} = \frac{f_{PLLClkout} \times 2}{D} \quad \text{Equation 2-2}$$

[Table 2-7](#) lists the allowable divider settings.

**Table 2-7: PLL Divider Settings**

Factor	Attribute	Valid Settings
M	PLL0_REFCLK_DIV PLL1_REFCLK_DIV	1, 2
N2	PLL0_FBDIV PLL1_FBDIV	1, 2, 3, 4, 5
N1	PLL0_FBDIV_45 PLL1_FBDIV_45	4, 5
D	RXOUT_DIV TXOUT_DIV	1, 2, 4, 8

## Ports and Attributes

[Table 2-8](#) and [Table 2-9](#) defines the ports and attributes for the PLL.

**Table 2-8: PLL Ports**

Port	Direction	Clock Domain	Description
PLL0LOCKDETCLK PLL1LOCKDETCLK	In	Clock	Stable reference clock for the detection of the feedback and reference clock signals to the PLL. The input reference clock to the PLL or any output clock generated from the PLL (e.g., TXOUTCLK) must not be used to drive this clock.  This clock is required only when using the PLL[0/1]FBCLKLOST and PLL[0/1]REFCLKLOST ports. It does not affect the PLL lock detection, reset, and power-down functions.
PLL0LOCKEN PLL1LOCKEN	In	Async	This port enables the PLL lock detector. It must always be tied High.
PLL0PD PLL1PD	In	Async	Active-High signal that powers down the PLL for power savings.
BGBYPASSB	In	Async	Reserved. This port must be set to 1'b1. This value should not be modified.
BGMONITORENB	In	Async	Reserved. This port must be set to 1'b1. This value should not be modified.
BGPDB	In	Async	Reserved. This port must be set to 1'b1. This value should not be modified.
BGRCALOVRD[4:0]	In	Async	Reserved. This port must be set to 5'b11111. This value should not be modified.
RCALENB	In	Async	Reserved. This port must be set to 1'b1. This value should not be modified.

Table 2-8: PLL Ports (Cont'd)

Port	Direction	Clock Domain	Description
PLL0REFCLKSEL[2:0] PLL1REFCLKSEL[2:0]	In	Async	<p>Input to dynamically select the input reference clock to the PLL. This input should be set to 3'b001 when only one clock source is connected to the PLL reference clock selection multiplexer.</p> <p>Reset must be applied to the PLL after changing the reference clock input.</p> <ul style="list-style-type: none"> <li>000: Reserved</li> <li>001: GTREFCLK0 selected</li> <li>010: GTREFCLK1 selected</li> <li>011: GTEASTREFCLK0 selected</li> <li>100: GTEASTREFCLK1 selected</li> <li>101: GTWESTREFCLK0 selected</li> <li>110: GTWESTREFCLK1 selected</li> <li>111: GTGREFCLK0 (PLL0) or GTGREFCLK1 (PLL1) selected</li> </ul>
PLL0RESET PLL1RESET	In	Async	This active-High port resets the dividers inside the PLL as well as the PLL lock indicator and status block.
PLL0FBCLKLOST PLL1FBCLKLOST	Out	PLL0LOCKDETCLK PLL1LOCKDETCLK	A High on this signal indicates the feedback clock from the PLL feedback divider to the phase frequency detector of the PLL is lost.
PLL0LOCK PLL1LOCK	Out	Async	This active-High PLL frequency lock signal indicates that the PLL frequency is within predetermined tolerance. The transceiver and its clock outputs are not reliable until this condition is met.
PLL0REFCLKLOST PLL1REFCLKLOST	Out	PLL0LOCKDETCLK PLL1LOCKDETCLK	A High on this signal indicates the reference clock to the phase frequency detector of the PLL is lost.

Table 2-9: PLL Attributes

Attribute	Type	Description
PLL0_CFG PLL1_CFG	27-bit Hex	Reserved. Configuration setting for the PLL. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
PLL0_FBDIV PLL1_FBDIV	Integer	PLL feedback divider setting as shown in <a href="#">Figure 2-10, page 36</a> . Valid settings are 1, 2, 3, 4, and 5.
PLL0_FBDIV_45 PLL1_FBDIV_45	Integer	PLL feedback divider settings as shown in <a href="#">Figure 2-10, page 36</a> . Valid settings are 4 and 5.
PLL0_LOCK_CFG PLL1_LOCK_CFG	9-bit Hex	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
PLL0_REFCLK_DIV PLL1_REFCLK_DIV	Integer	PLL reference clock divider M settings as shown in <a href="#">Figure 2-10, page 36</a> . Valid settings are 1 and 2.

**Table 2-9: PLL Attributes (Cont'd)**

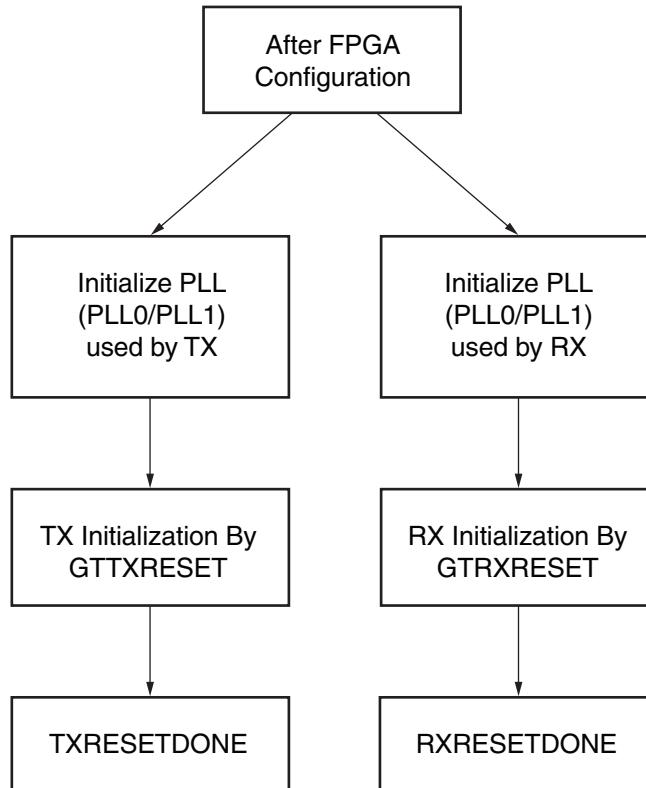
Attribute	Type	Description
PLL0_INIT_CFG PLL1_INIT_CFG	24-bit Hex	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
PLL0_DMON_CFG PLL1_DMON_CFG	1-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

## Reset and Initialization

The GTP transceiver must be initialized after FPGA device power-up and configuration before it can be used. The GTP transceiver's transmitter (TX) and receiver (RX) can be initialized independently and in parallel as shown in [Figure 2-11](#). The GTP transceiver's TX and RX initialization comprises two steps:

1. Initializing the associated PLL driving TX/RX
2. Initializing the TX and RX datapaths (PMA + PCS)

The GTP transceiver's TX and RX can receive a clock from either PLL0 or PLL1. The associated PLL (PLL0 /PLL1) used by the TX and RX must be initialized first before TX and RX initialization. Any PLL used by the TX and RX is reset individually and its reset operation is completely independent from all TX and RX resets. The TX and RX datapaths must be initialized only after the associated PLL is locked.



UG482\_c2\_15\_040412

**Figure 2-11: GTP Transceiver Initialization Overview**

The GTP transceiver's TX and RX use a state machine to control the initialization process. They are partitioned into a few reset regions. The partition allows the reset state machine to control the reset process in a sequence that the PMA can be reset first and the PCS can be reset after the assertion of the TXUSERRDY or RXUSERRDY. It also allows the PMA, the PCS, and functional blocks inside them to be reset individually when needed during normal operation.

The GTP transceiver offers two types of reset: initialization and component.

- **Initialization Reset:** This reset is used for complete GTP transceiver initialization. It must be used after device power-up and configuration. During normal operation, when necessary, GTTXRESET and GTRXRESET can also be used to reinitialize the GTP transceiver's TX and RX. GTTXRESET is the initialization reset port for the GTP transceiver TX. GTRXRESET is the initialization reset port for the GTP transceiver's RX.
- **Component Reset:** This reset is used for special cases and specific subsection resets while the GTP transceiver is in normal operation. TX component reset ports include TXPMARESET and TXPCSRESET. RX component reset ports include RXPMARESET, RXLPMRESET, EYESCANRESET, RXPCSRESET, RXBUFRESET, and RXOOBRESET.

For major coverage differences between initialization and component resets, refer to [Table 2-16](#) for the GTP transceiver's TX and [Table 2-20](#) and [Table 2-21](#) for the GTP transceiver's RX.

All reset ports described in this section initiate the internal reset state machine when driven High. The internal reset state machines are held in the reset state until these same reset ports are driven Low. These resets are all asynchronous. The guideline for the pulse width of these asynchronous resets is one period of the reference clock, unless otherwise noted.

**Note:** Reset ports should not be used for the purpose of power down. For details on proper power down usage, refer to [Power Down](#).

## Reset Modes

The GTP transceiver's RX resets can operate in two different modes: Sequential mode and single mode. The GTP transceiver's TX resets can operate only in sequential mode.

- Sequential mode: The reset state machine starts with an initialization or component reset input driven High and proceeds through all states after the requested reset states in the reset state machine, as shown in [Figure 2-13](#) for the GTP transceiver's TX or [Figure 2-18](#) for the GTP transceiver's RX until completion. The completion of sequential mode reset flow is signaled when (TX/RX)RESETDONE transitions from Low to High.
- Single mode: The reset state machine only executes the requested component reset independently for a predetermined time set by its attribute. It does not process any state after the requested state, as shown in [Figure 2-18](#) for the GTP transceiver's RX. The requested reset can be any component reset to reset the PMA, the PCS, or functional blocks inside them. The completion of a single mode reset is signaled when RXRESETDONE transitions from Low to High.

The GTP transceiver initialization reset must use sequential mode. All component resets can be operated in either sequential mode or single mode, except for TX resets, which can only operate in sequential mode.

The GTP transceiver uses GTRESETSEL to select between sequential reset mode and single reset mode. [Table 2-10](#) provides configuration details that apply to both the GTP transceiver's TX and RX. Reset modes have no impact on PLL0 or PLL1 resets. During normal operation, the GTP transceiver's TX or RX can be reset by applications in either sequential mode or single mode (GTP transceiver's RX only), which provides flexibility to reset a portion of the GTP transceiver. When using either sequential mode or single mode, RESETOVRD must be driven Low, as shown in [Table 2-10](#). RESETOVRD and GTRESETSEL must be set to the desired value 300–500 ns before the assertions of any reset.

**Table 2-10: GTP Transceiver Reset Modes Operation**

Operation Mode	RESETOVRD	GTRESETSEL
Sequential Mode	0	0
Single Mode	0	1

Table 2-11: GTP Transceiver Reset Mode Ports

Port	Dir	Clock Domain	Description
GTRESETSEL	In	Async	Reset mode enable port. Low: Sequential mode (recommended). High: Single mode.
RESETOVRD	In	Async	Reserved. Must be tied to ground.

## PLL Reset

The PLLs (PLL0 and PLL1) must be powered down using the PLL0PD and PLL1PD ports until reference clock edges are detected in the logic. After PLL0PD/PLL1PD is de-asserted, PLL0 and PLL1 must be reset before being used. Each GTPE2\_COMMON has six dedicated ports for PLL reset. As shown in Figure 2-12, PLL0RESET is an input that resets PLL0 and PLL1RESET is an input that resets PLL1. PLL0LOCK and PLL1LOCK are outputs that indicate that the reset process is done. The guideline for this asynchronous PLL0RESET and PLL1RESET pulse width is one period of the reference clock. The real PLL0 and PLL1 resets generated by the internal GTP transceiver circuits are much longer than the PLL0RESET and PLL1RESET High pulse duration. The time required for the PLL to lock is affected by a few factors, such as bandwidth setting and clock frequency.

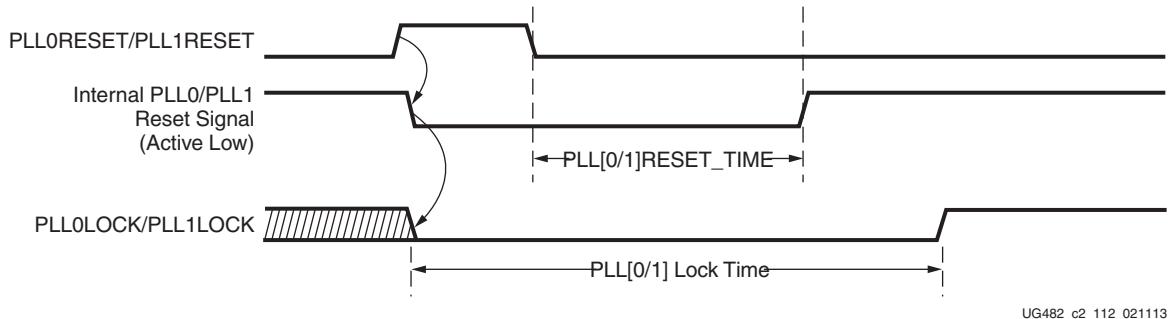


Figure 2-12: PLL Reset Timing Diagram

Table 2-12: PLL Reset Port

Port	Dir	Clock Domain	Description
PLL0RESET/ PLL1RESET	In	Async	This port is driven High and then deasserted to start the PLL reset.
PLL0LOCK/ PLL1LOCK	Out	Async	This active-High PLL frequency lock signal indicates that the PLL frequency is within a predetermined tolerance. The GTP transceiver and its clock outputs are not reliable until this condition is met.
PLL0LOCKEN/ PLL1LOCKEN	In	Async	This active-High signal enables the PLL lock detector.

Table 2-13: PLL Reset Attributes

Attribute	Type	Description
PLL[0/1]RESET_TIME (PLL[0/1]_INIT_CFG[9:0])	10-bit Binary	Reserved. Represents the time duration to apply internal PLL reset. Must be a non-zero value. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

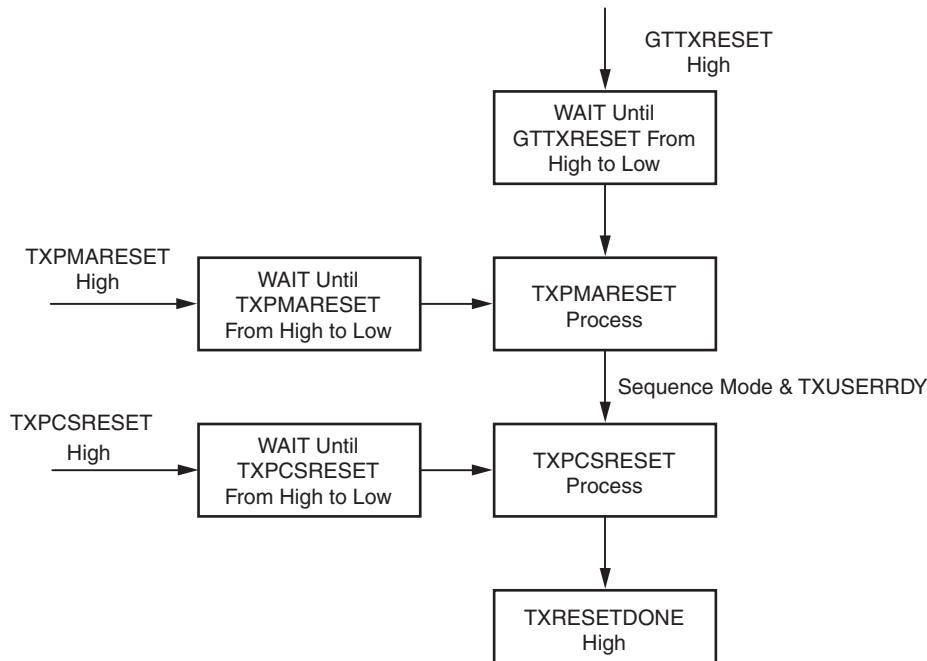
## TX Initialization and Reset

The GTP transceiver's TX uses a reset state machine to control the reset process. The GTP transceiver's TX is partitioned into two reset regions, TX PMA and TX PCS. The partition allows TX initialization and reset to be operated only in sequential mode, as shown in [Figure 2-13](#).

The initializing TX must use GTTXRESET in sequential mode. Activating the GTTXRESET input can automatically trigger a full asynchronous TX reset. The reset state machine executes the reset sequence, as shown in [Figure 2-13](#), covering the whole TX PMA and TX PCS. During normal operation, when needed, sequential mode allows the user to reset TX from activating TXPMARESET and continue the reset state machine until TXRESETDONE transitions from Low to High.

The TX reset state machine does not reset the PCS until TXUSERRDY is detected High. The user should drive TXUSERRDY High after these conditions are met:

1. All clocks used by the application including TXUSRCLK/TXUSRCLK2 are shown as stable or locked when the PLL or MMCM is used.
2. The user interface is ready to transmit data to the GTP transceiver.



UG482\_c2\_113\_020713

Figure 2-13: GTP Transceiver TX Reset State Machine Sequence

## Ports and Attributes

[Table 2-14](#) lists ports required by TX initialization process.

**Table 2-14: TX Initialization and Reset Ports**

Port	Dir	Clock Domain	Description
GTTXRESET	In	Async	This port is driven High and then deasserted to start the full TX reset sequence.
TXPMARESET	In	Async	This port is used to reset the TX PMA. It is driven High and then deasserted to start the TX PMA reset process. In sequential mode, activating this port resets both the TX PMA and the TX PCS.
TXPCSRESET	In	Async	This port is used to reset the TX PCS. It is driven High and then deasserted to start the PCS reset process. In sequential mode, activating this port only resets the TX PCS.
TXUSERRDY	In	Async	This port is driven High from the user's application when TXUSRCLK and TXUSRCLK2 are stable. For example, if an MMCM is used to generate both TXUSRCLK and TXUSRCLK2, then the MMCM lock signal can be used here.
TXRESETDONE	Out	TXUSRCLK2	This active-High signal indicates the GTP transceiver TX has finished reset and is ready for use. This port is driven Low when GTTXRESET goes High and is not driven High until the GTP transceiver TX detects TXUSERRDY High.
TXPMARESET DONE	Out	Async	This active-High signal indicates GTP TX PMA reset is complete. This port is driven Low when GTTXRESET or TXPMARESET is asserted.
CFGRESET	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
PCRSVDOOUT	Out	Async	Reserved.

[Table 2-15](#) lists attributes required by the GTP transceiver's TX initialization. In general cases, the reset time required by the TX PMA or the TX PCS varies depending on line rate. The factor affecting PMA reset time and PCS reset time are user-configurable attributes TXPMARESET\_TIME and TXPCSRESET\_TIME.

**Table 2-15: TX Initialization and Reset Attributes**

Attribute	Type	Description
TXPMARESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply a TX PMA reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when GTTXRESET or TXPMARESET is used to initiate the reset process.
TXPCSRESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply a TX PCS reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when TXPCSRESET is used to initiate the reset process.

## GTP Transceiver TX Reset in Response to Completion of Configuration

The TX reset sequence shown in [Figure 2-13](#) is not automatically started to follow global GSR. It must meet these conditions:

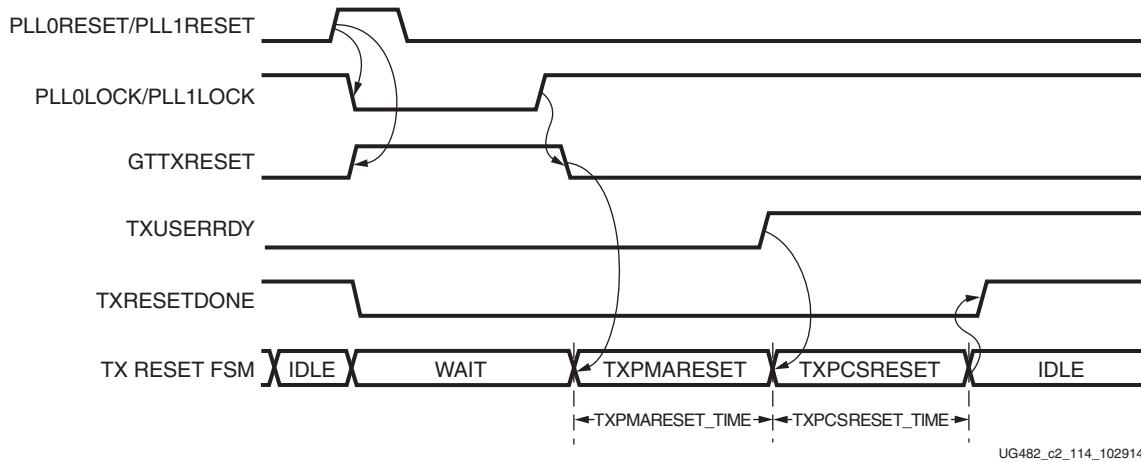
1. GTRESETSEL must be Low to use sequential mode.
2. GTTXRESET must be used.
3. TXPMARESET and TXPCSRESET must be constantly driven Low during the entire reset process before TXRESETDONE is detected High.
4. GTTXRESET cannot be driven Low until the associated PLL is locked.

If the reset mode is defaulted to sequential mode upon configuration, then PLL[0/1]RESET and GTTXRESET can be asserted after waiting for a minimum of 500 ns after configuration is complete.

If the reset mode is defaulted to single mode, then the user must:

1. Wait a minimum of 500 ns after configuration is complete.
2. Change reset mode to Sequential mode.
3. Wait another 300-500 ns.
4. Assert PLL[0/1]RESET and GTTXRESET.

It is recommended to use the associated PLLLOCK from either PLL0 or PLL1 to release GTTXRESET from High to Low as shown in [Figure 2-14](#). The TX reset state machine waits when GTTXRESET is detected High and starts the reset sequence when GTTXRESET is released Low.

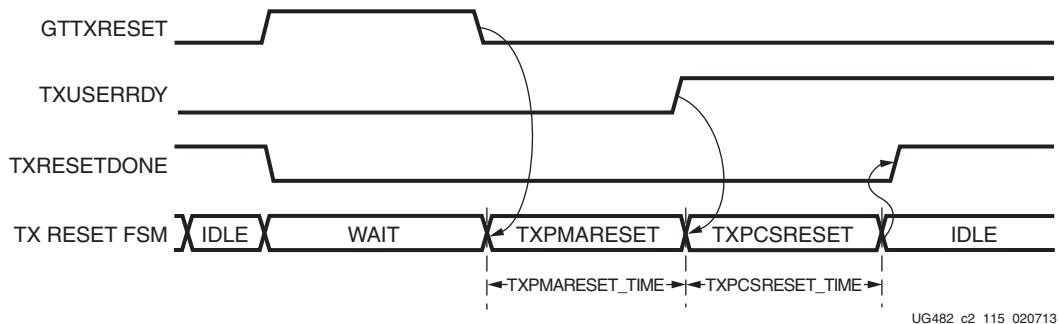


*Figure 2-14: GTP Transceiver Transmitter Initialization after FPGA Configuration*

### GTP Transceiver TX Reset in Response to GTTXRESET Pulse

The GTP transceiver allows the user to reset the entire TX completely at any time by sending GTTXRESET an active-High pulse. TXPMARESET\_TIME and TXPCSRESET\_TIME can be set statically or reprogrammed through DRP ports to adjust the required reset time before applying GTTXRESET. These conditions must be met when using GTTXRESET:

1. GTRESETSEL must be driven Low to use sequential mode.
2. TXPMARESET and TXPCSRESET must be driven constantly Low during the entire reset process before TXRESETDONE is detected High.
3. The associated PLL must indicate locked.
4. The guideline for this asynchronous GTTXRESET pulse width is one period of the reference clock.



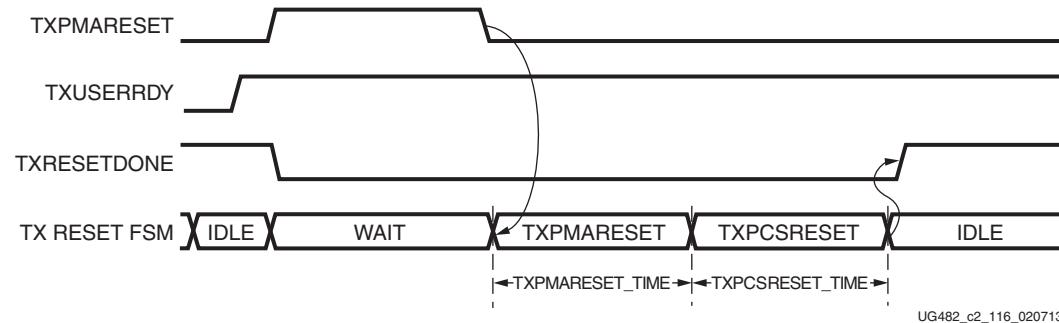
*Figure 2-15: GTP Transceiver Transmitter Reset after GTTXRESET Pulse*

### GTP Transceiver TX Component Reset

TX PMA and TX PCS can be reset individually. GTTXRESET must be driven constantly Low during the TXPMARESET or TXPCSRESET process before finish.

Driving TXPMARESET from High to Low starts the PMA reset process. TXPCSRESET must be driven constantly Low during the TXPMARESET process. In sequential mode

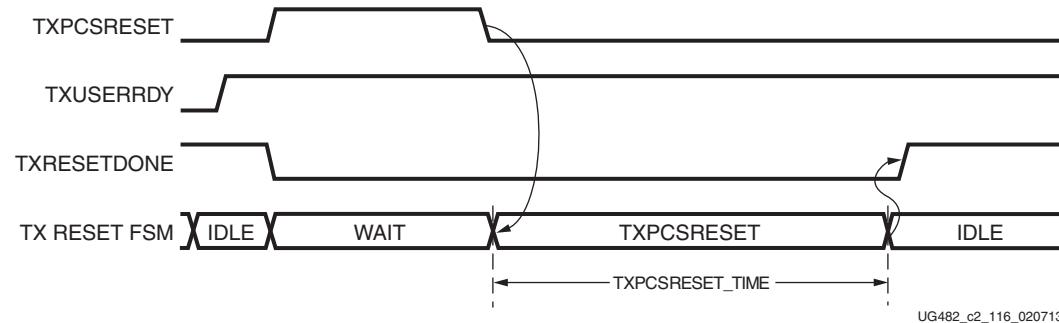
(Figure 2-16), the reset state machine automatically starts the PCS reset after finishing the PMA reset, if TXUSERRDY is High.



UG482\_c2\_116\_020713

**Figure 2-16: TXPMARESET in Sequential Mode**

Driving TXPCSRESET from High to Low starts the PCS reset process when TXUSERRDY is High. TXPMARESET must be driven constantly Low when the PCS is in reset process. In sequential mode, the reset state machine only resets the PCS (see Figure 2-17).



UG482\_c2\_116\_020713

**Figure 2-17: TXPCSRESET in Sequential Mode**

Table 2-16 summarizes all resets available to the GTP transceiver TX and components affected by them in sequential mode. Using TXPMARESET in sequential mode resets everything covered by GTTXRESET except the TX reset state machine.

**Table 2-16: TX Initialization Reset and Component Reset Coverage in Sequential Mode**

	Functional Blocks	GTTXRESET	TXPMARESET	TXPCSRESET
TX PCS	FPGA TX Fabric Interface	✓	✓	✓
	TX 8B/10B Encoder	✓	✓	✓
	TX Gearbox	✓	✓	✓
	TX Buffer	✓	✓	✓
	TX Pattern Generator	✓	✓	✓
	TX Polarity Control	✓	✓	✓
	TX Out-of-Band Signaling	✓	✓	✓
	TX Reset FSM	✓		

**Table 2-16: TX Initialization Reset and Component Reset Coverage in Sequential Mode (Cont'd)**

	Functional Blocks	GTTXRESET	TXPMARESET	TXPCSRESET
TX PMA	TX Configuration Driver	√	√	
	TX Receiver Detect for PCI Express Designs	√	√	
	TX PISO	√	√	

Table 2-17 lists the recommended resets for various situations.

**Table 2-17: Recommended Resets for Common Situations**

Situation	Components to be Reset	Recommended Reset <sup>(1)</sup>
After power up and configuration	Entire TX	GTTXRESET
After turning on a reference clock to the PLL being used	Entire TX	GTTXRESET
After changing the reference clock to the PLL being used	Entire TX	GTTXRESET
After assertion/deassertion of PLL[0/1]PD, for the PLL being used	Entire TX	GTTXRESET
After assertion/deassertion of TXPD[1:0]	Entire TX	GTTXRESET
TX rate change	TX PCS	Reset performed automatically
TX parallel clock source reset	TX PCS	TXPCSRESET
After entering or exiting far-end PMA loopback	Entire TX	GTTXRESET
After entering or exiting near-end PMA loopback	Entire RX	GTRXRESET

1. The recommended reset has the smallest impact on the other components of the GTP transceiver.

## After Power-up and Configuration

The entire GTP TX requires a reset after configuration. See [GTP Transceiver TX Reset in Response to Completion of Configuration](#) for the procedure.

## After Turning on a Reference Clock to the PLL Being Used

If the reference clock(s) changes or the GTP transceiver(s) are powered up after configuration, GTTXRESET should be toggled after the PLL fully completes its reset procedure.

## After Changing the Reference Clock to the PLL Being Used

Whenever the reference clock input to the PLL is changed, the PLL must be reset afterwards to ensure that it locks to the new frequency. The GTTXRESET should be toggled after the PLL fully completes its reset procedure.

## After Assertion/Deassertion of PLL[0/1]PD, for the PLL Being Used

When the PLL being used goes back to normal operation after power down, the PLL must be reset. The GTTXRESET should be toggled after the PLL fully completes its reset procedure.

## After Assertion/Deassertion of TXPD[1:0]

After the TXPD signal is deasserted, GTTXRESET must be toggled.

## TX Rate Change

When a rate change is performed using the TXRATE port and TXRATEMODE is set to 1'b0, the required reset sequence is performed automatically. When TXRATEDONE is asserted, it indicates that both the rate change and the necessary reset sequence have been applied and completed.

If the TX buffer is enabled, the TXBUF\_RESET\_ON\_RATE\_CHANGE attribute should be set to TRUE to allow the TX buffer to reset automatically after the rate change.

If TX buffer bypass mode is used, alignment must be repeated after TXRATEDONE is asserted.

## TX Parallel Clock Source Reset

The clocks driving TXUSRCLK and TXUSRCLK2 must be stable for correct operation.

These clocks are often driven from an MMCM in the FPGA to meet phase and frequency requirements. If the MMCM loses lock and begins producing incorrect output, TXPCSRESET should be toggled after the clock source re-locks.

If TX buffer bypass mode is used, alignment must be repeated after the completion of the reset procedure.

## RX Initialization and Reset

The GTP transceiver's RX uses a reset state machine to control the reset process. Due to its complexity, the GTP transceiver's RX is partitioned into more reset regions than the GTP transceiver's TX. The partition allows RX initialization and reset to be operated in either sequential mode or single mode as shown in [Figure 2-18](#):

### 1. RX in Sequential Mode

To initialize the GTP transceiver's RX, GTRXRESET must be used in sequential mode. Activating the GTRXRESET input can automatically trigger a full asynchronous RX reset. The reset state machine executes the reset sequence as shown in [Figure 2-18](#), covering the entire RX PMA and RX PCS. During normal operation, sequential mode also allows the user to initiate a reset by activating any of these resets including RXPMARESET, RXLPMRESET, EYESCANRESET, RXPCSRESET, and RXBUFRESET, and continue the reset state machine until RXRESETDONE transitions from Low to High.

### 2. RX in Single Mode

When the GTP transceiver's RX is in single mode, RXPMARESET, RXLPMRESET, EYESCANRESET, RXPCSRESET, and RXBUFRESET in the reset sequence can be executed individually and independently without triggering a reset on other reset regions.

In either sequential mode or single mode, the RX reset state machine does not reset the PCS until RXUSERRDY goes High. The user should drive RXUSERRDY High after these conditions are met:

1. All clocks used by the application, including RXUSRCLK and RXUSRCLK2, are shown to be stable or locked when the PLL or the MMCM is used.
2. The user interface is ready to receive data from the GTP transceiver.

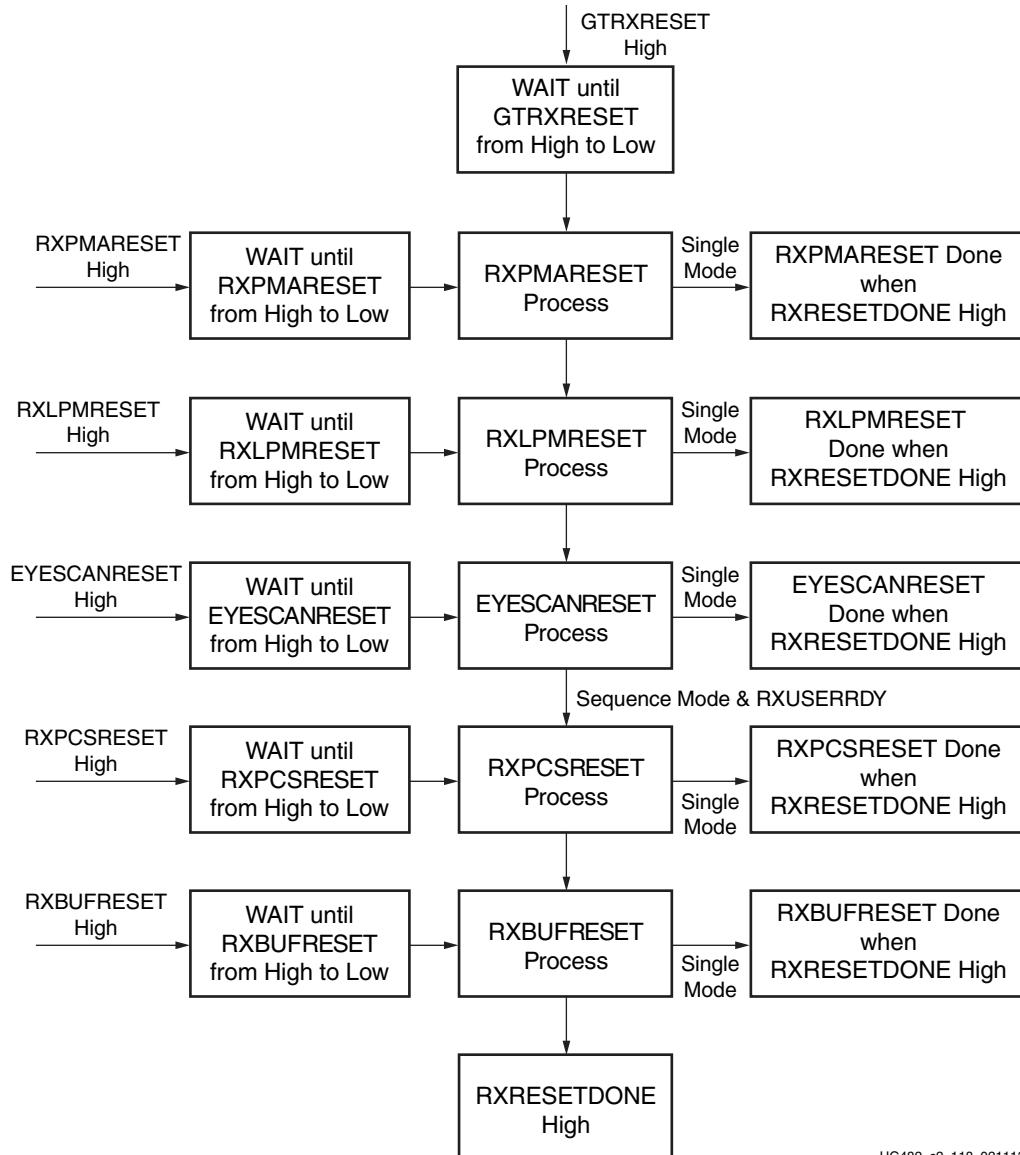


Figure 2-18: GTP Transceiver RX Reset State Machine Sequence

## Ports and Attributes

**Table 2-18** lists the ports required by the GTP transceiver's RX initialization process.

**Table 2-18: RX Initialization and Reset Ports**

Port	Dir	Clock Domain	Description
GTRXRESET	In	Async	This port is driven High and then deasserted to start the full Channel RX reset sequence.
RXOSCALRESET	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTDONE	Out	Async	Reserved.
RXPMARESET	In	Async	This port is driven High and then deasserted to start RX PMA reset process. In single mode, activating RXPMARESET resets only the RX PMA blocks not including CDR and LPM. In sequential mode, activating RXPMARESET starts the RX reset process as shown in <a href="#">Figure 2-18</a> from RXPMARESET and followed by RXCDRPHASEREST, RXCDRFREQRESET, RXLPMRESET, EYESCANRESET, RXPCSRESET, and RXBUFRESET. Detailed coverage on sequential mode is listed in <a href="#">Table 2-20</a> .
RXCDRRESET	In	Async	Reserved. Tied Low.
RXCDRFREQRESET	In	Async	Reserved. Tied Low.
RXLPMRESET	In	Async	This port is driven High and then deasserted to start the LPM reset process. In single mode, activating RXLPMRESET resets only the RX LPM circuits. In sequential mode, activating RXLPMRESET starts the RX reset process as shown in <a href="#">Figure 2-18</a> from RXLPMRESET and followed by EYESCANRESET, RXPCSRESET, and RXBUFRESET. Detailed coverage in sequential mode is listed in <a href="#">Table 2-20</a> .
EYESCANRESET	In	Async	This port is driven High and then deasserted to start the EYESCAN reset process. In single mode, activating EYESCANRESET resets only the RX Eye Scan circuits. In sequential mode, activating EYESCANRESET starts the RX reset process as shown in <a href="#">Figure 2-18</a> from EYESCANRESET and followed by RXPCSRESET, and RXBUFRESET. Detailed coverage in sequential mode is listed in <a href="#">Table 2-20</a> .

Table 2-18: RX Initialization and Reset Ports (Cont'd)

Port	Dir	Clock Domain	Description
RXPCSRESET	In	Async	This port is driven High and then deasserted to start the PCS reset process. In single mode, activating RXPCSRESET resets only the RX PCS circuits. In sequential mode, activating RXPCSRESET starts the RX reset process as shown in <a href="#">Figure 2-18</a> from RXPCSRESET and followed by RXBUFRESET. Detailed coverage in sequential mode is listed in <a href="#">Table 2-20</a> .  In both modes, RXPCSRESET does not start the reset process until RXUSERRDY is High.
RXBUFFRESET	In	Async	This port is driven High and then deasserted to start the RX elastic buffer reset process. In either single mode or sequential mode, activating RXBUFFRESET resets the RX elastic buffer only.
RXUSERRDY	In	Async	This port is driven High from the user's application when RXUSRCLK and RXUSRCLK2 are stable. For example, if an MMCM is used to generate both RXUSRCLK and RXUSRCLK2, then the MMCM lock signal can be used here.
RXRESETDONE	Out	RXUSRCLK2	When asserted, this active-High signal indicates the GTP transceiver's RX has finished reset and is ready for use. This port is driven Low when GTRXRESET is driven High. This signal is not driven High until RXUSERRDY goes High.
RXPMARESETDONE	Out	Async	This active-High signal indicates GTP RX PMA reset is complete. This port is driven Low when GTRXRESET or RXPMARESET is asserted.
RXOOBRESET	In	Async	This port can be used to reset the OOB individually. It should be tied Low if the OOB function is not used or the OOB single reset is not required.  RXOOBRESET is independent from the GTP transceiver's RX reset state machine sequence as shown in <a href="#">Figure 2-18</a> . Sequential mode and single mode do not apply to RXOOBRESET.  Activating RXOOBRESET does not cause RXRESETDONE to transition from Low to High or High to Low.

[Table 2-19](#) lists the attributes required by GTP transceiver's RX initialization. In general cases, the reset time required by each reset on the RX datapath varies depending on line

rate and function. The factors affecting each reset time are user-configurable attributes listed in [Table 2-19](#).

**Table 2-19: RX Initialization and Reset Attributes**

Attribute	Type	Description
RXOSCALRESET_TIME	5-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when GTRXRESET is used to initiate the reset process.
RXOSCALRESET_TIMEOUT	5-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when GTRXRESET is used to initiate the reset process.
RXPMARESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply the RX PMA reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when using GTRXRESET or RXPMARESET to initiate reset process.
RXCDRPHRESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply RX CDR Phase reset. Must be a non-zero value when using RXCDRRESET to initialize the reset process.
RXCDRFREQRESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply the RX CDRFREQ reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when using RXCDRFREQRESET to initiate the reset process.
RXLPMRESET_TIME	7-bit Binary	Reserved. Represents the time duration to apply the RX LPM reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when using RXLPMRESET to initiate the reset process.
RXISCANRESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply the RX EYESCAN reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when using EYESCANRESET_TIME to initiate the reset process.

Table 2-19: RX Initialization and Reset Attributes (Cont'd)

Attribute	Type	Description
RXPCSRESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply the RX PCS reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when using RXPCSRESET to initiate the reset process.
RXBUFFRESET_TIME	5-bit Binary	Reserved. Represents the time duration to apply the RX BUFFER reset. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. Must be a non-zero value when using RXBUFFRESET to initiate the reset process.

## GTP Transceiver RX Reset in Response to Completion of Configuration

The RX reset sequence shown in [Figure 2-18](#) is not automatically started following the global GSR. These conditions must be met:

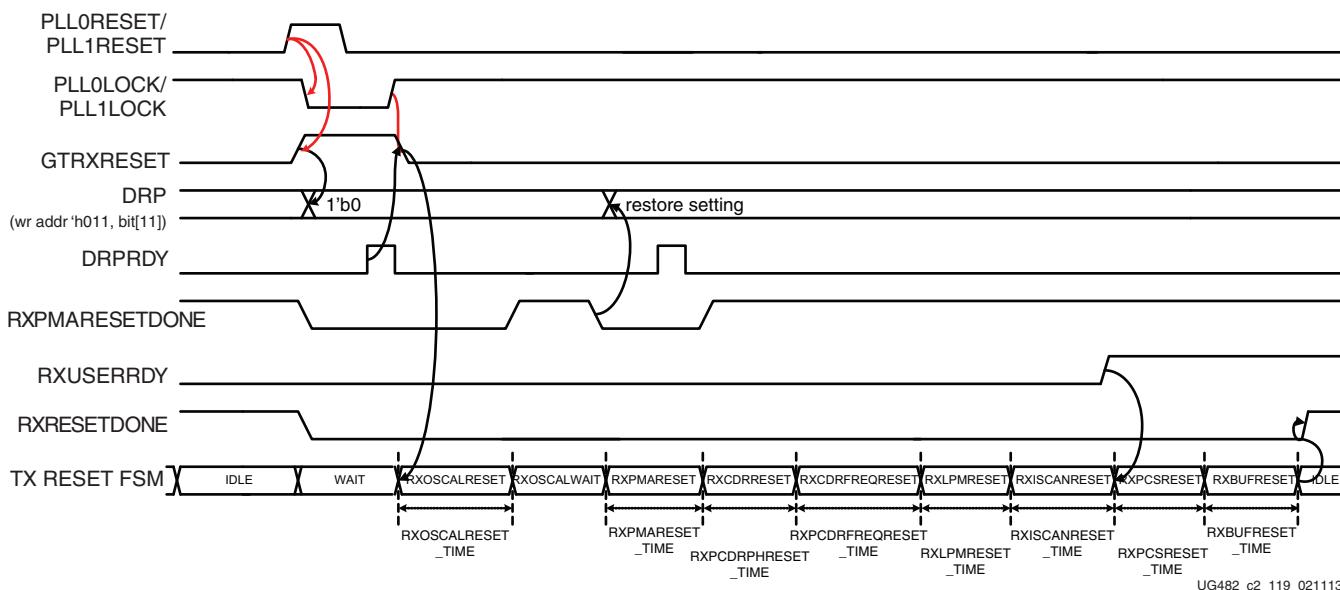
1. GTRESETSEL must be driven Low to use the sequential mode.
2. GTRXRESET must be used.
3. All single reset inputs including RXPMARESET, RXCDRRESET, RXCDRFREQRESET, RXLPMRESET, EYSCANRESET, RXPCSRESET, and RXBUFRESET must be constantly held Low during the entire reset process before RXRESETDONE goes High.
4. GTRXRESET cannot be driven Low until the associated PLL is locked.

If the reset mode is defaulted to sequential mode upon configuration, then PLL[0/1]RESET and GTRXRESET can be asserted after waiting for a minimum of 500 ns after configuration is complete.

If the reset mode is defaulted to single mode, then the user must:

1. Wait a minimum of 500 ns after configuration is complete.
2. Change reset mode to Sequential mode.
3. Wait another 300-500 ns.

When users want to issue an GTRXRESET upon configuration, the steps in [Figure 2-19](#) should be performed.



**Figure 2-19: GTP Transceiver Receiver after FPGA Configuration**

Notes relevant to [Figure 2-19](#):

1. “DRP wr” denotes the function of performing a DRP write to addr 9’ h011. The exact DRP transaction is not shown.
2. The sequence of events in [Figure 2-19](#) is not drawn to scale.
3. When the user wants to trigger RX reset upon configuration, assert and release PLL[0/1]RESET while GTRXRESET is kept asserted. The assertion of GTRXRESET causes RXPMARESETDONE to go Low.

4. Issue a DRP write to the GTPE2\_CHANNEL primitive, DRPADDR 9'h011, set bit[11] to 1'b0.
  - a. To ensure only bit[11] of DRPADDR 9'h011 is modified, it is best to perform a read-modify-write function.
5. Upon DRP write completion, the user can set and hold GTRXRESET Low as desired. The user can extend the assertion of GTRXRESET, as long as GTRXRESET is held High until the DRP write is completed.

**Note:** It is recommended to use the associated PLLLOCK from either the PLL0 or PLL1 to release GTRXRESET from High to Low as shown in [Figure 2-19](#).
6. Wait for the falling edge of RXPMARESETDONE.
7. Issue a DRP write to the GTPE2\_CHANNEL primitive, DRPADDR 9'h011, restoring the original setting for bit[11]. The completion of this DRP write must occur before RXPMARESETDONE switches from Low to High. RXPMARESETDONE stays Low for a minimum of 0.66 µs.
8. GTRXRESET should be driven with an output of a register to avoid glitches.
9. RXPMARESET\_TIME should be set to 5'h3. This should be the default setting.
10. The sequence above will simulate correctly if SIM\_RESET\_SPEEDUP is set to FALSE. If SIM\_RESET\_SPEEDUP is set to TRUE, the above sequence should be bypassed.

## GTP Transceiver RX Reset in Response to GTRXRESET Pulse

The GTP transceiver allows the user to completely reset the entire GTP transceiver's RX at any time when needed by sending GTRXRESET an active High pulse. All RX reset attributes listed in [Table 2-18](#) can be set statically or reprogrammed through DRP ports to adjust the required reset time before applying GTRXRESET. These conditions must be met to use GTRXRESET:

1. GTRESETSEL must be driven Low to use sequential mode.
2. All reset inputs shown on the left of [Figure 2-18](#) including RXPMARESET, RXCDRRESET, RXCDRFREQRESET, RXLPMRESET, EYESCANRESET, RXPCSRESET, and RXBUFRESET must be constantly driven Low during the entire reset process before RXRESETDONE is detected High.
3. The associated PLL must indicate locked.
4. The steps for issuing GTRXRESET is illustrated in [Figure 2-20](#).

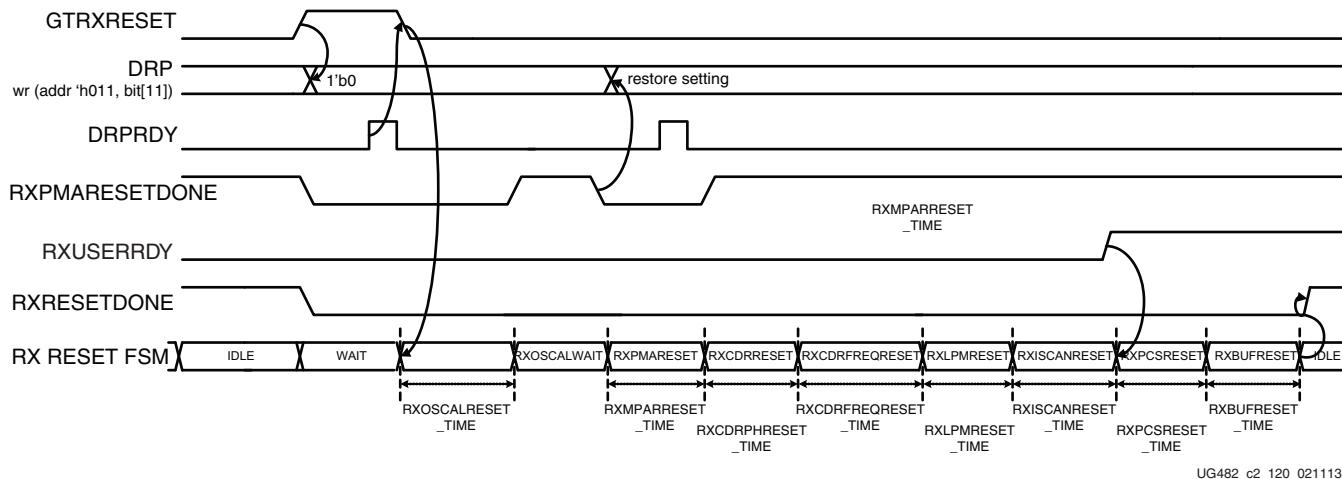


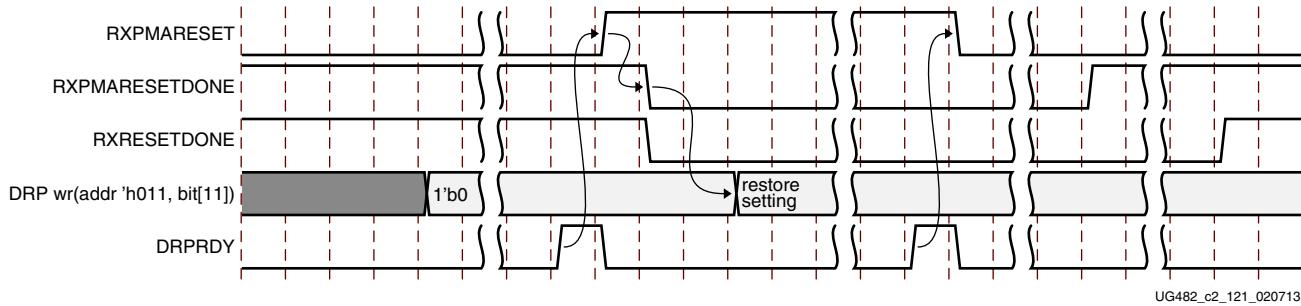
Figure 2-20: GTP Transceiver Receiver Reset after GTRXRESET Pulse

Notes relevant to Figure 2-20:

1. “DRP wr” denotes the function of performing a DRP write to addr 9 ‘ h011. The exact DRP transaction is not shown.
  2. The sequence of events in Figure 2-20 is not drawn to scale.
  3. When the user wants to trigger RX reset upon configuration, assert and release PLL[0/1]RESET while GTRXRESET is kept asserted. The assertion of GTRXRESET causes RXPMARESETDONE to go Low.
  4. Issue a DRP write to the GTPE2\_CHANNEL primitive, DRPADDR 9 ‘ h011, set bit[11] to 1 ‘ b0.
    - a. To ensure only bit[11] of DRPADDR 9 ‘ h011 is modified, it is best to perform a read-modify-write function.
  5. Upon DRP write completion, the user can set and hold GTRXRESET Low as desired. The user can extend the assertion of GTRXRESET, as long as GTRXRESET is held High until the DRP write is completed.
  6. Wait for the falling edge of RXPMARESETDONE.
  7. Issue a DRP write to the GTPE2\_CHANNEL primitive, DRPADDR 9 ‘ h011, restoring the original setting for bit[11]. The completion of this DRP write must occur before RXPMARESETDONE switches from Low to High. RXPMARESETDONE stays Low for a minimum of 0.66  $\mu$ s.
  8. GTRXRESET should be driven with an output of a register to avoid glitches.
  9. RXPMARESET\_TIME should be set to 5 ‘ h3. This should be the default setting.
- The sequence above will simulate correctly if SIM\_RESET\_SPEEDUP is set to FALSE. If SIM\_RESET\_SPEEDUP is set to TRUE, the above sequence should be bypassed.

## GTP Transceiver RX PMA Reset

When users want to issue an RXPMARESET, the steps in [Figure 2-21](#) should be performed.



**Figure 2-21: GTP Transceiver RXPMARESET Sequence**

1. “DRP wr” denotes the function of performing a DRP write to addr 9 ‘h011. The exact DRP transaction is not shown.
2. The sequence of events in [Figure 2-21](#) is not drawn to scale.
3. When the user wants to trigger a RXPMARESET, issue a DRP write to the GTPE2\_CHANNEL primitive, DRPADDR 9 ‘h011, set bit[11] to 1 ‘b0.
  - a. To ensure only bit[11] of DRPADDR 9 ‘h011 is modified, it is best to perform a read-modify-write function.
4. Upon DRP write completion, set and hold RXPMARESET High.
5. Wait for RXPMARESETDONE to be detected Low.
6. Issue a DRP write to the GTPE2\_CHANNEL primitive, DRPADDR 9 ‘h011, restoring the original setting for bit[11].
7. Upon DRP write completion, the user can set and hold RXPMARESET Low as desired. The user can extend the assertion of RXPMARESET, as long as RXPMARESET is held High until the DRP write is completed.
8. RXPMARESET should be driven with an output of a register to avoid glitches.

## GTP Transceiver RX Component Resets

GTP transceiver RX component resets can operate in either sequential mode or single mode. They are primarily used for special cases. These resets are needed when only a specific subsection needs to be reset. [Table 2-20](#) and [Table 2-21](#) also summarize all resets available to the GTP transceiver’s RX and components affected by them in both sequential mode and single mode. These resets are all asynchronous.

Table 2-20: RX Component Reset Coverage in Sequential Mode

	Functional Blocks	GTRX RESET	RXPMA RESET	RXLPM RESET	EYESCAN RESET	RXPCS RESET	RXBPF RESET
RX PCS	FPGA RX Fabric Interface	✓	✓	✓	✓	✓	
	RX Gearbox	✓	✓	✓	✓	✓	
	RX Status Control	✓	✓	✓	✓	✓	
	RX Elastic Buffer Delay Aligner	✓	✓	✓	✓	✓	
	RX 8B/10B Encoder	✓	✓	✓	✓	✓	
	RX Comma Detect and Alignment	✓	✓	✓	✓	✓	
	RX Polarity	✓	✓	✓	✓	✓	
	PRBS Checker	✓	✓	✓	✓	✓	
	RX Elastic Buffer	✓	✓	✓	✓	✓	✓
	RX Reset FSM	✓					
RX PMA	RX Analog Front End	✓	✓				
	RX Out-of-Band Signaling	✓	✓				
	RX SIPO	✓	✓				
	RX CDR Phase Path	✓	✓				
	RX CDR Frequency Path	✓	✓				
	RX LPM	✓	✓	✓			
	RX ISCAN	✓	✓	✓	✓		

Table 2-21: RX Component Reset Coverage in Single Mode

	Functional Blocks	GTRX RESET	RXPMA RESET	RXLPM RESET	EYESCAN RESET	RXPCS RESET	RXBIF RESET	RXOOB RESET
RX PCS	FPGA RX Fabric Interface					✓		
	RX Gearbox					✓		
	RX Status Control					✓		
	RX Delay Aligner					✓		
	RX 8B/10B Encoder					✓		
	RX Comma Detect and Alignment					✓		
	RX Polarity					✓		
	PRBS Checker					✓		
	RX Elastic Buffer						✓	
RX Reset FSM								
RX PMA	RX Analog Front End		✓					
	RX Out-of-Band Signaling		✓					✓
	RX SIPO		✓					
	RX CDR Phase Path							
	RX CDR Frequency Path							
	RX LPM			✓				
	RX ISCAN				✓			

Table 2-22 lists the recommended resets for various situations.

Table 2-22: Recommended Resets for Common Situations

Situation	Components to be Reset	Recommended Reset <sup>(1)</sup>
After power up and configuration	Entire RX	GTRXRESET
After turning on a reference clock to the PLL being used	Entire RX	GTRXRESET
After changing the reference clock to the PLL being used	Entire RX	GTRXRESET
After assertion/deassertion of PLL[0/1]PD, for the PLL being used	Entire RX	GTRXRESET
After assertion/deassertion of RXPD[1:0]	Entire RX	GTRXRESET
RX rate change	Entire RX	GTRXRESET or reset sequence is performed automatically due to RXRATE
RX parallel clock source reset	RX PCS	RXPCSRESET

Table 2-22: Recommended Resets for Common Situations (Cont'd)

Situation	Components to be Reset	Recommended Reset <sup>(1)</sup>
After remote power up	Entire RX	GTRXRESET
Electrical idle	Entire RX	Handled automatically with appropriate attribute settings
After connecting RXN/RXP <sup>(2)</sup>	Entire RX	GTRXRESET
After recovered clock becomes stable	RX Elastic Buffer	RXBUFFRESET
After an RXBUFFER error	RX Elastic Buffer	RXBUFFRESET
After changing channel bonding mode on the fly	RX Elastic Buffer	RX elastic buffer is reset automatically after change in channel bonding mode by setting RXBUF_RESET_ON_CB_CHANGE to TRUE
After PRBS error	PRBS Error Counter	PRBSCNTRESET
After comma realignment	RX Elastic Buffer (optional)	RX elastic buffer is reset automatically after comma realignment by setting RXBUF_RESET_ON_COMMALIGN to TRUE
After entering or exiting Far-End PMA loopback	Entire TX	GTIXRESET
After entering or exiting Near-End PMA loopback	Entire RX	GTRXRESET
<b>Notes:</b>		
1. The recommended reset has the smallest impact on the other components of the GTP transceiver.		
2. It is assumed that RXN/RXP are connected simultaneously.		

## After Power-up and Configuration

The entire GTP RX requires a reset after configuration. See [GTP Transceiver RX Reset in Response to Completion of Configuration](#) for procedure.

## After Turning on a Reference Clock to the PLL Being Used

If the reference clock(s) changes or GTP transceiver(s) are powered up after configuration, GTRXRESET should be toggled after the PLL fully completes its reset procedure.

## After Changing the Reference Clock to the PLL Being Used

Whenever the reference clock input to the PLL is changed, the PLL must be reset afterwards to ensure that it locks to the new frequency. The GTRXRESET should be toggled after the PLL fully completes its reset procedure.

## After Assertion/Deassertion of PLL[0/1]PD, for the PLL Being Used

When the PLL being used goes back to normal operation after power down, the PLL must be reset. The GTRXRESET should be toggled after the PLL fully completes its reset procedure.

## After Assertion/Deassertion of RXPD[1:0]

After the RXPD signal is deasserted, GTRXRESET must be toggled.

## RX Rate Change

In most cases, in addition to changing the output divider, an RX rate change requires changing the RX CDR loop filter settings via DRP (See [RX CDR](#)). After writing in the proper RX CDR loop filter setting and updating the RX\_OUTDIV attribute via DRP, the RX must be reset by toggling the GTRXRESET port. In cases where the CDR loop filter does not need to be updated via DRP, rate change can be performed using the RXRATE port with RXRATEDMODE set to 1'b0. As a result, the required reset sequence is performed automatically. When RXRATEDONE is asserted in response to RXRATE, it indicates both rate change and the necessary reset sequence has been applied and completed.

If RX buffer is enabled, RXBUF\_RESET\_ON\_RATE\_CHANGE attribute should be set to "TRUE" to allow RX buffer to reset automatically after rate change when using the RXRATE port. If RX buffer bypass mode is used, alignment must be repeated after RXRATEDONE is asserted.

## RX Parallel Clock Source Reset

The clocks driving RXUSRCLK and RXUSRCLK2 must be stable for correct operation. These clocks are often driven from an MMCM in the FPGA to meet phase and frequency requirements. If the MMCM loses lock and begins producing incorrect output, RXPCSRESET should be toggled after the clock source relocks. If RX buffer bypass mode is used, alignment must be repeated after the completion of the reset procedure.

## After Remote Power-Up

If the source of incoming data is powered up after the GTP transceiver that is receiving its data is operating, the RX side must be reset to ensure a clean lock to the incoming data

## Electrical Idle Reset

For protocols that support OOB and electrical idle, when the differential voltage of the RX input to the transceiver drops to OOB or electrical idle levels, the RX CDR will be managed automatically when attributes associated with electrical idle are set to appropriate values. Recommended values from the 7 Series FPGA Transceivers Wizard should be used.

## After Connecting RXN/RXP

When the RX data to the GTP transceiver comes from a connector that can be plugged in and unplugged, the RX side must be reset when the data source is plugged in to ensure that it can lock to incoming data.

## After Recovered Clock Becomes Stable

Depending on the design of the clocking scheme, it is possible for the RX reset sequence to be completed before the CDR is locked to the incoming data. In this case, the recovered clock may not be stable when RXRESETDONE is asserted. When RX buffer is used, RXBUFRRESET should be triggered after the recovered clock becomes stable. When RX buffer bypass is used, the alignment procedure should not start until the recovered clock becomes stable. Refer to [DS181, Artix-7 FPGAs Data Sheet](#) for successful CDR lock-to-data criteria.

## After an RX Elastic Buffer Error

After an RX elastic buffer overflow or underflow, the RX elastic buffer must be reset using RXBUFRESET to ensure correct behavior.

## After Changing Channel Bonding Mode on the Fly

When set to TRUE, RXBUF\_RESET\_ON\_CB\_CHANGE enables automatic reset of the RX elastic buffer when the RXCHANBONDMASTER, RXCHANBONDMASTER, or RXCHANBONDLEVEL change.

## After a PRBS Error

PRBSCNTRESET is asserted to reset the PRBS error counter.

## After Comma Realignment

When set to TRUE, RXBUF\_RESET\_ON\_COMMAALIGN enables automatic reset of the RX elastic buffer during comma realignment.

# Power Down

## Functional Description

The GTP transceiver supports a range of power-down modes. These modes support both generic power management capabilities as well as those defined in the PCI Express® and SATA standards.

The GTP transceiver offers different levels of power control. Each channel in each direction can be powered down separately using TXPD and RXPD. The PLL0PD port directly affects the PLL0 while the PLL1PD port affects PLL1.

## Ports and Attributes

[Table 2-23](#) defines the power-down ports.

*Table 2-23: Power-Down Ports*

Port	Dir	Clock Domain	Description
PLL0PD	In	Async	This active-High signal powers down PLL0.
PLL1PD	In	Async	This active-High signal powers down PLL1.
RXPD[1:0]	In	Async	Powers down the RX lane according to the PCI Express PIPE protocol encoding. 00: P0 (normal operation) 01: P0s (low recovery time power down) 10: P1 (longer recovery time) 11: P2 (lowest power state)

Table 2-23: Power-Down Ports (*Cont'd*)

Port	Dir	Clock Domain	Description
TXPD[1:0]	In	TXUSRCLK2 (TXPDELECIDLEMODE makes this port asynchronous)	Powers down the TX lane according to the PCI Express PIPE protocol encoding. 00: P0 (normal operation) 01: P0s (low recovery time power down) 10: P1 (longer recovery time; Receiver Detection still on) 11: P2 (lowest power state) Attributes can control the transition times between these power-down states.
TXPDELECIDLEMODE	In	Async	Determines if TXELECIDLE and TXPD should be treated as synchronous or asynchronous signals.
TXPHDLYPD	In	Async	TX phase and delay alignment circuit power down. It is set to 1 'b0 in TX buffer bypass mode. 0: Power up the TX phase and delay alignment circuit. 1: Power down the TX phase and delay alignment circuit.
RXPHDLYPD	In	Async	RX phase and delay alignment circuit power down. It is set to 1 'b0 in RX buffer bypass mode. 0: Power up the RX phase and delay alignment circuit. 1: Power down the RX phase and delay alignment circuit.

Table 2-24 defines the power-down attributes.

**Table 2-24: Power-Down Attributes**

Attribute	Type	Description
PD_TRANS_TIME_FROM_P2	12-bit Hex	Counter settings for programmable transition time from P2 state for PCIe. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
PD_TRANS_TIME_NONE_P2	8-bit Hex	Counter settings for programmable transition time to/from all states except P2 for PCIe. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
PD_TRANS_TIME_TO_P2	8-bit Hex	Counter settings for programmable transition time to P2 state for PCIe. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TRANS_TIME_RATE	8-bit Hex	Counter settings for programmable transition time when the rate is changed using the [TX/RX]RATE pins for all protocols including the PCIe protocol (Gen2/Gen1 data rates). The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RX_CLKMUX_PD	1-bit Binary	The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TX_CLKMUX_PD	1-bit Binary	The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

## Generic Power-Down Capabilities

The GTP transceiver provides several power-down features that can be used in a wide variety of applications. [Table 2-25](#) summarizes these capabilities.

**Table 2-25: Basic Power-Down Functions Summary**

Function	Controlled By	Affects
PLL0 Control	PLL0PD	Powers down PLL0.
PLL1 Control	PLL1PD	Powers down PLL1.
TX Power Control	TXPD[1:0]	The TX of the GTP transceiver.
RX Power Control	RXPD[1:0]	The RX of the GTP transceiver.

## PLL Power Down

To activate the PLL0 power-down mode, the active-High PLL0PD signal is asserted. Similarly, to activate the PLL1 power-down mode, the active-High PLL1PD signal is asserted. When either PLL0PD or PLL1PD is asserted, the corresponding PLL is powered down. As a result, all clocks derived from the respective PLL are stopped.

During initial configuration and power-on, PLL0/PLL1 must be powered down using the PLL0PD/PLL1PD port until reference clock edges are detected. PLL0/PLL1 should be powered down if the reference clock stops. For PLL0-based designs, when PLL1 is not used, PLL1PD can be tied High. For PLL1-based designs, when PLL0 is not used, the PLL0PD can be tied High.

Recovery from this power state is indicated by the assertion of the corresponding PLL lock signal.

## TX and RX Power Down

When the TX and RX power control signals are used in non PCI Express implementations, TXPD and RXPD can be used independently. Also, when these interfaces are used in non PCI Express applications, only two power states are supported, as shown in [Table 2-26](#). When using this power-down mechanism, these must be true:

- TXPD[1] and TXPD[0] are connected together.
- RXPD[1] and RXPD[0] are connected together.
- TXDETECTRX must be strapped Low.
- TXELECIDLE must be strapped to TXPD[1] and TXPD[0].

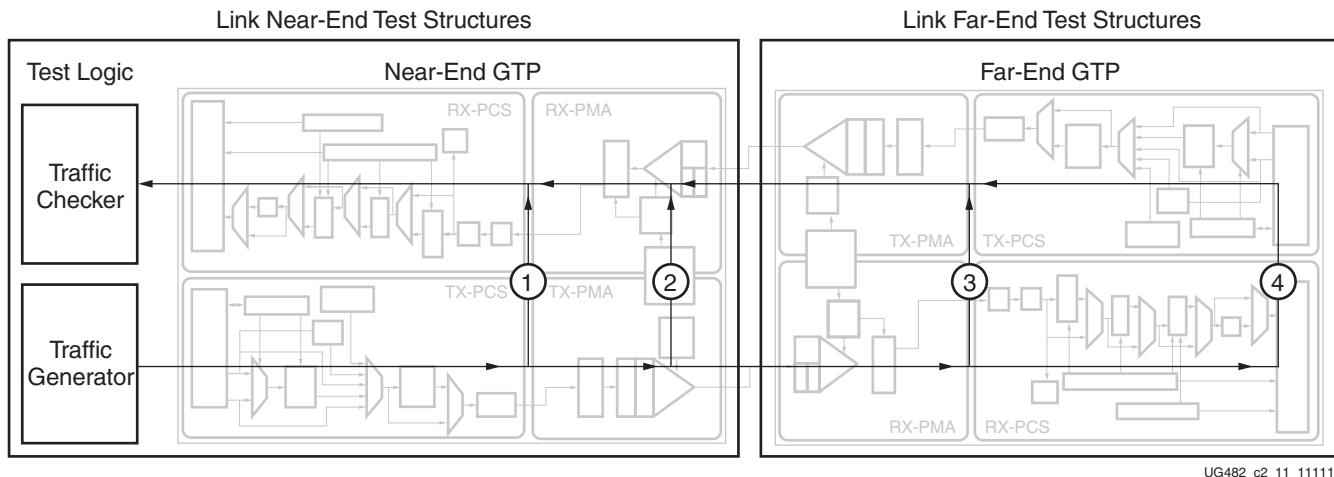
**Table 2-26: TX and RX Power States for Operation that are not for PCI Express Designs**

TXPD[1:0] or RXPD[1:0]	Description
00	Normal mode. Transceiver TX or RX is active sending or receiving data.
11	Power-down mode. Transceiver TX or RX is idle.

## Loopback

### Functional Description

Loopback modes are specialized configurations of the transceiver datapath where the traffic stream is folded back to the source. Typically, a specific traffic pattern is transmitted and then compared to check for errors. [Figure 2-22](#) illustrates a loopback test configuration with four different loopback modes.



**Figure 2-22: Loopback Testing Overview**

Loopback test modes fall into two broad categories:

- Near-end loopback modes loop transmit data back in the transceiver closest to the traffic generator.
- Far-end loopback modes loop received data back in the transceiver at the far end of the link.

Loopback testing can be used either during development or in deployed equipment for fault isolation. The traffic patterns used can be either application traffic patterns or specialized pseudo-random bit sequences. Each GTP transceiver has a built-in PRBS generator and checker.

Each GTP transceiver features several loopback modes to facilitate testing:

- Near-end PCS Loopback (path 1 in [Figure 2-22](#))

The RX elastic buffer must be enabled and RX\_XCLK\_SEL must be set to RXREC for Near-end PCS loopback to function properly.

While in Near-end PCS loopback, the RX XCLK domain is clocked by the TX PMA parallel clock (TX XCLK). If the RXOUTCLK is used to clock FPGA logic and RXOUTCLKSEL is set to RXOUTCLKPMA during normal operation, one of the following two items must be changed when placing the GTP transceiver into Near-end PCS Loopback.

1.) Set RXOUTCLKSEL to select RXOUTCLKPCS

or

2.) Set RXCDRHOLD = 1 'b1

- Near-end PMA Loopback (path 2 in [Figure 2-22](#))

A GTRXRESET is required after entering and exiting Near-end PMA loopback.

- Far-end PMA Loopback (path 3 in [Figure 2-22](#))

The TX buffer must be enabled and TX\_XCLK\_SEL must be set to TXOUT for Far-end PMA loopback to function properly.

While in Far-end PMA loopback, the write side of the TX buffer is clocked by the RX PMA parallel clock (RX XCLK).

A GTTXRESET is required after entering and exiting Far-end PMA loopback.

- Far-end PCS Loopback (path 4 in [Figure 2-22](#))

If clock correction is not used, a transceiver in Far-end PCS loopback must use the same reference clock used by the transceiver that is the source of the loopback data. Regardless of whether clock correction is used or not, the ports TXUSRCLK and RXUSRCLK must be driven by the same clocking resource (BUFG, BUFH).

Far-end PCS loopback is not supported when both or either gearbox in the channel is enabled.

## Ports and Attributes

[Table 2-27](#) and [Table 2-28](#) define the loopback ports and attributes, respectively.

*Table 2-27: Loopback Ports*

Port	Dir	Clock Domain	Description
LOOPBACK[2:0]	In	Async	000: Normal operation 001: Near-End PCS Loopback 010: Near-End PMA Loopback 011: Reserved 100: Far-End PMA Loopback 101: Reserved 110: Far-End PCS Loopback

*Table 2-28: Loopback Attributes*

Attribute	Type	Description
LOOPBACK_CFG	1-bit Binary	Reserved.
PMA_LOOPBACK_CFG	1-bit Binary	Reserved.

# Dynamic Reconfiguration Port

## Functional Description

The dynamic reconfiguration port (DRP) allows the dynamic change of parameters of the GTPE2\_CHANNEL and GTPE2\_COMMON primitives. The DRP interface is a processor-friendly synchronous interface with an address bus (DRPADDDR) and separated data buses for reading (DRPDO) and writing (DRPDI) configuration data to the primitives. An enable signal (DRPEN), a read/write signal (DRPWE), and a ready/valid signal (DRPRDY) are the control signals that implement read and write operations, indicate operation completion, or indicate the availability of data.

## Ports and Attributes

[Table 2-29](#) shows the DRP related ports for GTPE2\_CHANNEL.

**Table 2-29: DRP Ports of GTPE2\_CHANNEL**

Port	Dir	Clock Domain	Description
DRPADDDR[8:0]	In	DRPCLK	DRP address bus.
DRPCLK	In	N/A	DRP interface clock.
DRPEN	In	DRPCLK	DRP enable signal. 0: No read or write operation performed. 1: Enables a read or write operation.  For write operations, DRPWE and DRPEN must be driven High for one DRPCLK cycle only (see <a href="#">Figure 2-23</a> for correct operation). For read operations, DRPEN must be driven High for one DRPCLK cycle only (see <a href="#">Figure 2-24</a> for correct operation).
DRPDI[15:0]	In	DRPCLK	Data bus for writing configuration data from the FPGA logic resources to the transceiver.
DRPRDY	Out	DRPCLK	Indicates operation is complete for write operations and data is valid for read operations. See <a href="#">Figure 2-23</a> and <a href="#">Figure 2-24</a> for the assertion of DRPRDY signal after a write and a read operation.
DRPDO[15:0]	Out	DRPCLK	Data bus for reading configuration data from the GTP transceiver to the FPGA logic resources.
DRPWE	In	DRPCLK	DRP write enable. 0: Read operation when DRPEN is 1. 1: Write operation when DRPEN is 1.  For write operations, DRPWE and DRPEN should be driven High for one DRPCLK cycle only. See <a href="#">Figure 2-23</a> for correct operation.

[Table 2-30](#) shows the DRP related ports for GTPE2\_COMMON.

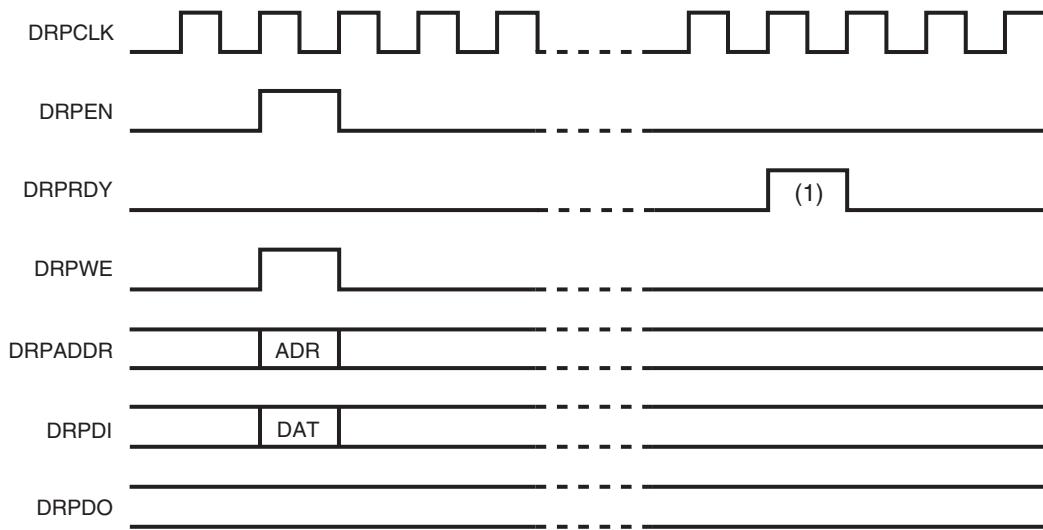
Table 2-30: DRP Ports of GTPE2\_COMMON

Port	Dir	Clock Domain	Description
DRPADDR[7:0]	In	DRPCLK	DRP address bus.
DRPCLK	In	N/A	DRP interface clock.
DRPEN	In	DRPCLK	DRP enable signal. 0: No read or write operation performed. 1: Enables a read or write operation. For write operations, DRPWE and DRPEN must be driven High for one DRPCLK cycle only (see <a href="#">Figure 2-23</a> for correct operation). For read operations, DRPEN must be driven High for one DRPCLK cycle only (see <a href="#">Figure 2-24</a> for correct operation).
DRPDI[15:0]	In	DRPCLK	Data bus for writing configuration data from the FPGA logic resources to the transceiver.
DRPRDY	Out	DRPCLK	Indicates operation is complete for write operations and data is valid for read operations.
DRPDO[15:0]	Out	DRPCLK	Data bus for reading configuration data from the GTP transceiver to the FPGA logic resources.
DRPWE	In	DRPCLK	DRP write enable. 0: Read operation when DRPEN is 1. 1: Write operation when DRPEN is 1. For write operations, DRPWE and DRPEN must be driven High for one DRPCLK cycle only. Please see <a href="#">Figure 2-23</a> for correct operation.

## Usage Model

### Write Operation

Figure 2-23 shows the DRP write operation timing. New DRP operation can be initiated when DRPRDY is asserted.



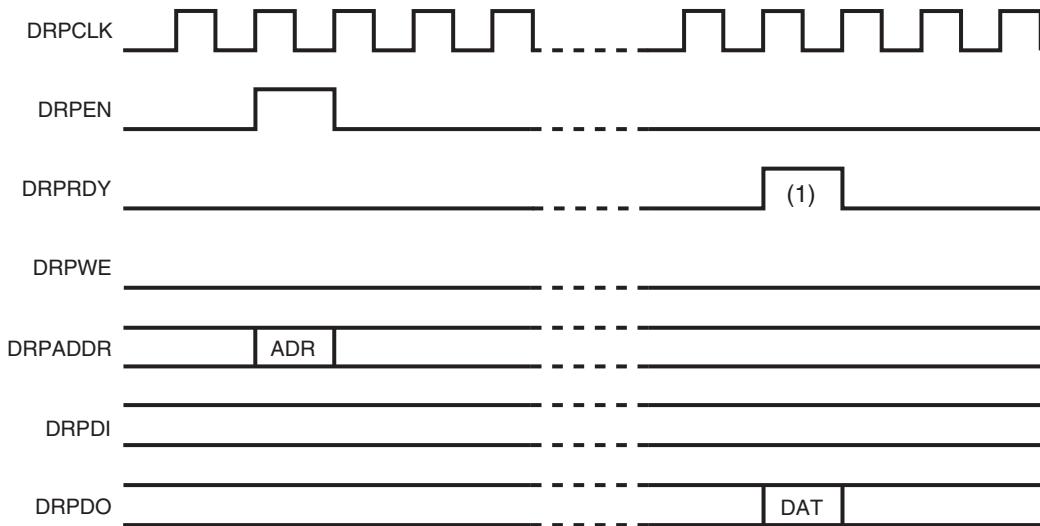
(1) After a DRP write is requested, it takes 5 DRPCLK clock cycles for the DRPRDY signal to be asserted.

UG482\_c2\_12\_040213

Figure 2-23: DRP Write Timing

### Read Operation

Figure 2-24 shows the DRP read operation timing. New DRP operation can be initiated when DRPRDY is asserted.



(1) After a DRP read is requested, it takes:

- R/W registers: 5 DRPCLK clock cycles for the DRPRDY signal to be asserted.
- Read-only registers: Ceiling((DRPCLK freq/USRCLK freq)\*6)+7 DRPCLK clock cycles for the DRPRDY signal to be asserted. The duration depends on the ratio between the DRP clock frequency and the USRCLK clock frequency.

UG482\_c2\_13\_040213

**Figure 2-24: DRP Read Timing**

## Digital Monitor

### Functional Description

The receiver uses an adaptive algorithm in optimizing a link. The digital monitor provides visibility into the convergence state of these adaptation loops. The digital monitor requires a clock; RXUSRCLK2 can be used for this. The select line on which adaptation loops to monitor is controlled through the RX\_DEBUG\_CFG attribute. The output port DMONITOROUT contains the convergence code(s) for a selected loop. All loops are continuous. A continuous loop has three possible convergence states: min, max, or dithering.

## Ports and Attributes

[Table 2-31](#) shows the digital monitor ports.

**Table 2-31: Digital Monitor Ports**

Port	Dir	Clock Domain	Description
DMONITOROUT[14:0]	Out	Async/Local Clock	Digital Monitor Output Bus: • [14:8] Unused • [7] - Internal Clock Adaptation loops: • [6:0] RXOS • [6:3] RXLPMHF, RXLPMLF
DMONITORCLK	In	Async	Digital monitor clock
DMONFIFORESET	In	DMONITORCLK	Reserved. Tie To GND. Reset use for sync mode operation.

[Table 2-32](#) shows the digital monitor attributes.

**Table 2-32: Digital Monitor Attributes**

Attribute	Type	Description
RX_DEBUG_CFG[13:0]	14-bit Binary	[13:8] - Reserved. Set to 6'h00 [7:6] - Reserved. Set to 2'b11. [5] - Reserved. Set to 1'b0. [4:0] - Select adaptation loop: See <a href="#">Table 2-33</a> .
CFOK_CFG[42]	1-bit Binary	Reserved. Set to 1'b1.
DMONITOR_CFG[23:0]	24-bit Binary	Reserved. Set to 24'h008101.

**Table 2-33: Select Adaptation Loop Description Details**

DRP Address	DRP DI	Loop Description	Code Mapping
0x0A5	0x00C2	RXLPMOS -Base line wander cancellation 7-bit signed with double neutral	7'd0 - min (neg) 7'63 - neutral 7'64 - neutral 7'127 - max (pos)
0x0A5	0x00C3	RXLPMHF - LPM high-frequency gain	4'd0 - min 4'd15 - max
0x0A5	0x00C4	RXLPMLF - LPM low-frequency gain	4'd0 - min 4'd15 - max

## Use Mode

Reading loop values out of the digital monitor requires a clock on input clock port DMONITORCLK, change adaptation loop select through DRP, and monitor output DMONITOROUT. Set the DMONITOR\_CFG attribute via the DRP port to the appropriate loop for monitoring. The DRP location of DMONITOR\_CFG is:

```
0x086[15:0] = DMONITOR_CFG[15:0]
```

```
0x087[7:0] = DMONITOR_CFG[23:16]
```

The output can be observed on DMONITOROUT. The signals from the digital monitor are LSB aligned and asynchronous.

### Capturing the Digital Monitor Output

The DMONITOROUT signals change slowly in comparison to the RXUSRCLK2s. One way to capture the DMONITOROUT is described here.

```
reg [7:0] compare1, compare2, dmonitorout_sync;
always@ (posedge RXUSRCLK2)
begin
    if (reset)
        begin
            compare1 <= 8'd0;
            compare2 <= 8'd0;
            dmonitorout_sync <= 8'd0;
        end
    else
        begin
            compare1 <= DMONITOROUT;
            compare2 <= compare1;

            if (compare1 == compare2)
                dmonitorout_sync <= compare2;
            else
                dmonitorout_sync<=dmonitorout_sync;

        end //else
    end //always
```

Any method that captures the information successfully is valid.

### Capturing the Digital Monitor Output Through Software

The dmonitorout\_sync described in the Verilog code in the section above can be mapped into host processor memory to capture the digital monitor output. The channel DRP port can be mapped into host processor memory to select the adaptation loop to be monitored.

The following example C code is provided as an illustration. The drpread and drpwrite functions are DRP operations described in [Usage Model, page 71](#). The captureDMON function reads dmonitorout\_sync register described in the above Verilog code.

```
///////////////
// Function Prototypes
///////////////

void drpwrite(unsigned int drpaddress, unsigned int drpvalue);
unsigned int drpread(unsigned int drpaddress);
```

```
unsigned int captureDMON(unsigned int msb, unsigned int lsb);
///////////
///////////////////////////////
// Initialize Digital Monitor
///////////////////////////////
// Write CFOK_CFG[41] Attribute
drpwrite(0x08B, 0x0490);
// Write DMONITOR_CFG[23:0]
drpwrite(0x087, 0x0000);
drpwrite(0x086, 0x8101);

///////////////////////////////
// Read Digital Monitor as often as required
///////////////////////////////

while(!done) {

    // RXOS
    drpwrite(0x0A5, 0x00C2);
    captureDMON(6, 0);

    ///////////////////////////////
    // LPM Mode Only
    //////////////////////////////

    // LPM Mode Only: RXLPMHF
    drpwrite(0x0A5, 0x00C3);
    captureDMON(6, 3);

    // LPM Mode Only: RXLPMLF
    drpwrite(0x0A5, 0x00C4);
    captureDMON(6, 3);}

}
```

### Interpreting the Digital Monitor Output

This section describes which bits of the DMONITOROUT bus are relevant for the appropriate selection of DMON\_CFG and the manner of interpreting the output.

- RXLPMOS[6:0] = DMONITOROUT[6:0]  
7'd0 = -Full scale  
7'd63, 7'd64 = 0  
7'd127 = +Full scale
- RXLPMHF [3:0] = RXLPMLF [3:0] = DMONITOROUT[6:3]  
4'd0 = 0  
4'd15 = Full scale



# Transmitter

## TX Overview

### Functional Description

This chapter shows how to configure and use each of the functional blocks inside the transmitter (TX). Each transceiver includes an independent transmitter, which consists of a PCS and a PMA. Figure 3-1 shows the functional blocks of the transmitter. Parallel data flows from the FPGA logic into the FPGA TX interface, through the PCS and PMA, and then out the TX driver as high-speed serial data.

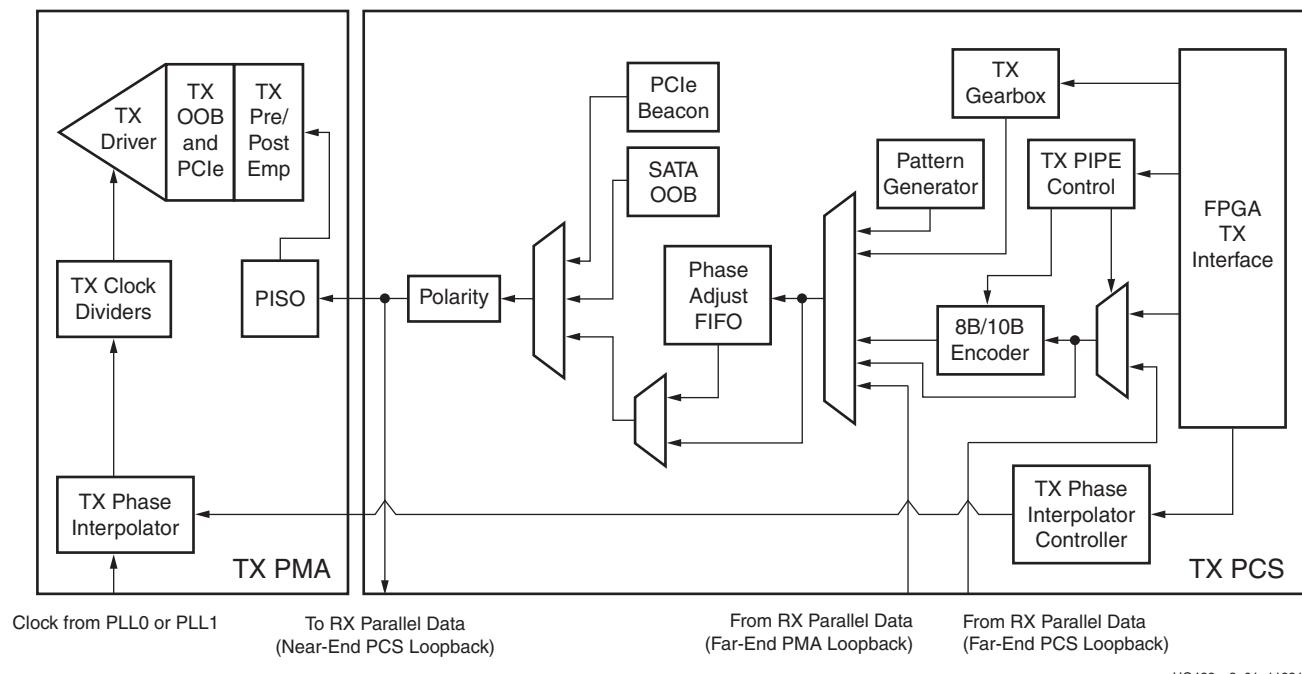


Figure 3-1: GTP Transceiver TX Block Diagram

The key elements within the GTP transceiver TX are:

1. [FPGA TX Interface, page 78](#)
2. [TX 8B/10B Encoder, page 85](#)
3. [TX Gearbox, page 88](#)
4. [TX Buffer, page 95](#)

5. TX Pattern Generator, page 105
6. TX Pattern Generator, page 105
7. TX Polarity Control, page 108
8. TX Fabric Clock Output Control, page 109
9. TX Configurable Driver, page 116
10. TX Receiver Detect Support for PCI Express Designs, page 123
11. TX Out-of-Band Signaling, page 125

## FPGA TX Interface

### Functional Description

The FPGA TX interface is the FPGA's gateway to the TX datapath of the GTP transceiver. Applications transmit data through the GTP transceiver by writing data to the TXDATA port on the positive edge of TXUSRCLK2. The width of the port can be configured to be two or four bytes wide. The actual width of the port depends on the TX\_DATA\_WIDTH attribute and TX8B10BEN port setting. Port widths can be 16, 20, 32, and 40 bits. The rate of the parallel clock (TXUSRCLK2) at the interface is determined by the TX line rate, the width of the TXDATA port, and whether or not 8B/10B encoding is enabled. A second parallel clock (TXUSRCLK) must be provided for the internal PCS logic in the transmitter. This section shows how to drive the parallel clocks and explains the constraints on those clocks for correct operation.

### Interface Width Configuration

The 7 series FPGA GTP transceiver contains a 2-byte internal datapath. The FPGA interface width is configurable by setting the TX\_DATA\_WIDTH attribute. When the 8B/10B encoder is enabled, the TX\_DATA\_WIDTH attribute must be configured to 20 bits or 40 bits, and in this case, the FPGA TX interface only uses the TXDATA ports. For example, TXDATA[15:0] is used when the FPGA interface width is 16. When the 8B/10B encoder is bypassed, the TX\_DATA\_WIDTH attribute can be configured to any of the available widths: 16, 20, 32, or 40 bits.

[Table 3-1](#) shows how the interface width for the TX datapath is selected. 8B/10B encoding is described in more detail in [TX 8B/10B Encoder, page 85](#).

**Table 3-1: FPGA TX Interface Datapath Configuration**

TX8B10BEN	TX_DATA_WIDTH	FPGA Interface Width	Internal Data Width
1	20	16	20
	40	32	20
0	16	16	16
	20	20	20
	32	32	16
	40	40	20

When the 8B/10B encoder is bypassed and the TX\_DATA\_WIDTH is 20 or 40, the TXCHARDISPMODE and TXCHARDISPVAL ports are used to extend the TXDATA port

from 16 to 20 bits, or 32 to 40 bits. [Table 3-2](#) shows the data transmitted when the 8B/10B encoder is disabled. When the TX gearbox is used, refer to [TX Gearbox, page 88](#) for data transmission order.

**Table 3-2: TX Data Transmitted when 8B/10B Encoder Bypassed**

	< < < Data Transmission Order is Right to Left (LSB to MSB) < < <																																						
	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Data Transmitted	TXCHARDISPMODE[3]	TXCHARDISPVAL[3]	TXDATA[31:24]	TXCHARDISPMODE[2]	TXCHARDISPVAL[2]	TXDATA[23:16]	TXCHARDISPMODE[1]	TXCHARDISPVAL[1]	TXDATA[15:8]	TXCHARDISPMODE[0]	TXCHARDISPVAL[0]	TXDATA[7:0]																											

### TXUSRCLK and TXUSRCLK2 Generation

The FPGA TX interface includes two parallel clocks: TXUSRCLK and TXUSRCLK2. TXUSRCLK is the internal clock for the PCS logic in the GTP transceiver transmitter. The required rate for TXUSRCLK depends on the internal datapath width of the GTPE2\_CHANNEL primitive and the TX line rate of the GTP transceiver transmitter. [Equation 3-1](#) shows how to calculate the required rate for TXUSRCLK.

$$\text{TXUSRCLK Rate} = \frac{\text{Line Rate}}{\text{Internal Datapath Width}} \quad \text{Equation 3-1}$$

TXUSRCLK2 is the main synchronization clock for all signals into the TX side of the GTP transceiver. Most signals into the TX side of the GTP transceiver are sampled on the positive edge of TXUSRCLK2. TXUSRCLK2 and TXUSRCLK have a fixed-rate relationship based on the TX\_DATA\_WIDTH setting. [Table 3-3](#) shows the relationship between TXUSRCLK2 and TXUSRCLK per TX\_DATA\_WIDTH value.

**Table 3-3: TXUSRCLK2 Frequency Relationship to TXUSRCLK**

FPGA Interface Width	TX_DATA_WIDTH	TXUSRCLK2 Frequency
2-Byte	16, 20	$F_{\text{TXUSRCLK2}} = F_{\text{TXUSRCLK}}$
4-Byte	32, 40	$F_{\text{TXUSRCLK2}} = F_{\text{TXUSRCLK}}/2$

These rules about the relationships between clocks must be observed for TXUSRCLK and TXUSRCLK2:

- TXUSRCLK and TXUSRCLK2 must be positive-edge aligned, with as little skew as possible between them. As a result, low-skew clock resources (BUFGs and BUFHs) should be used to drive TXUSRCLK and TXUSRCLK2.
- Even though they might run at different frequencies, TXUSRCLK, TXUSRCLK2, and the transmitter reference clock must have the same oscillator as their source. Thus TXUSRCLK and TXUSRCLK2 must be multiplied or divided versions of the transmitter reference clock.

### Ports and Attributes

[Table 3-4](#) defines the FPGA TX Interface ports.

Table 3-4: FPGA TX Interface Ports

Port	Dir	Clock Domain	Description
TXCHARDISPMODE[3:0]	In	TXUSRCLK2	When 8B/10B encoding is disabled, TXCHARDISPMODE is used to extend the data bus for 20- and 40-bit TX interfaces.
TXCHARDISPVAL[3:0]	In	TXUSRCLK2	When 8B/10B encoding is disabled, TXCHARDISPVAL is used to extend the data bus for 20- and 40-bit TX interfaces.
TXDATA[31:0]	In	TXUSRCLK2	The bus for transmitting data. The width of this port depends on TX_DATA_WIDTH: TX_DATA_WIDTH = 16, 20: TXDATA[15:0] = 16 bits wide TX_DATA_WIDTH = 32, 40: TXDATA[31:0] = 32 bits wide  When a 20-bit or 40-bit bus is required, the TXCHARDISPVAL and TXCHARDISPMODE ports from the 8B/10B encoder is concatenated with the TXDATA port. See <a href="#">Table 3-2, page 79</a> .
TXUSRCLK	In	Clock	This port is used to provide a clock for the internal TX PCS datapath.
TXUSRCLK2	In	Clock	This port is used to synchronize the FPGA logic with the TX interface. This clock must be positive-edge aligned to TXUSRCLK when TXUSRCLK is provided by the user.

[Table 3-5](#) defines the FPGA TX interface attributes.

Table 3-5: FPGA TX Interface Attributes

Attribute	Type	Description
TX_DATA_WIDTH	Integer	Sets the bit width of the TXDATA port. When 8B/10B encoding is enabled, TX_DATA_WIDTH must be set to 20 or 40. Valid settings are 16, 20, 32, and 40. See <a href="#">Interface Width Configuration, page 78</a> for more information.

## Using TXOUTCLK to Drive the TX Interface

Depending on the TXUSRCLK and TXUSRCLK2 frequencies, there are different ways FPGA clock resources can be used to drive the parallel clock for the TX interface. [Figure 3-2](#) through [Figure 3-5](#) show different ways FPGA clock resources can be used to drive the parallel clocks for the TX interface. In these examples, the TXOUTCLK is derived from the MGTREFCLK0[P/N] or MGTREFCLK1[P/N] and the TXOUTCLKSEL = 011 to select the TXPLLREFCLK\_DIV1 path as indicated in [Figure 3-20, page 110](#).

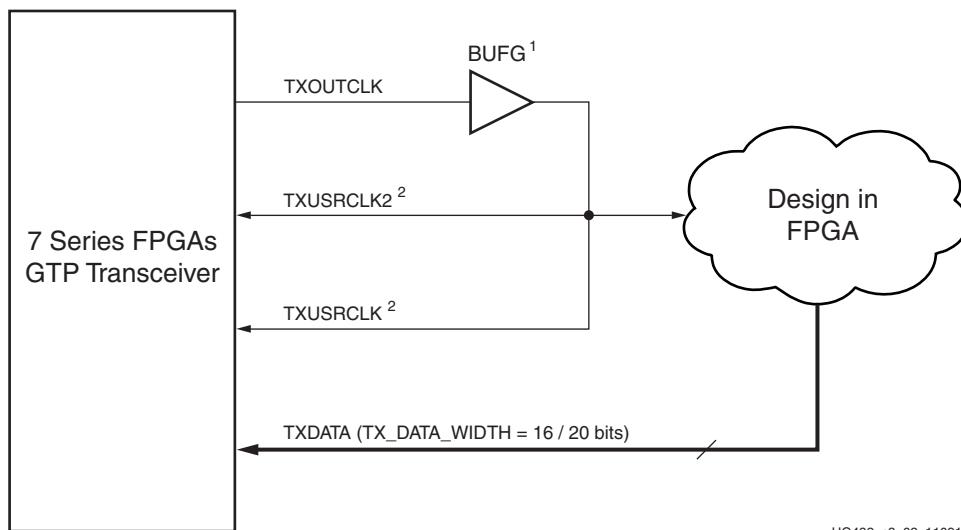
- Depending on the input reference clock frequency and the required line rate, an MMCM and the appropriate TXOUTCLKSEL port setting is required. The

CORE Generator™ tool creates a sample design based on different design requirements for most cases.

- In use models where TX buffer is bypassed, there are additional restrictions on the clocking resources. Refer to [TX Pattern Generator, page 105](#) for more information.

### TXOUTCLK Driving GTP Transceiver TX in 2-Byte Mode

In [Figure 3-2](#), TXOUTCLK is used to drive TXUSRCLK and TXUSRCLK2 for 2-byte mode (TX\_DATA\_WIDTH = 16 or 20) in a single-lane configuration. The frequency of TXUSRCLK2 is equal to TXUSRCLK.



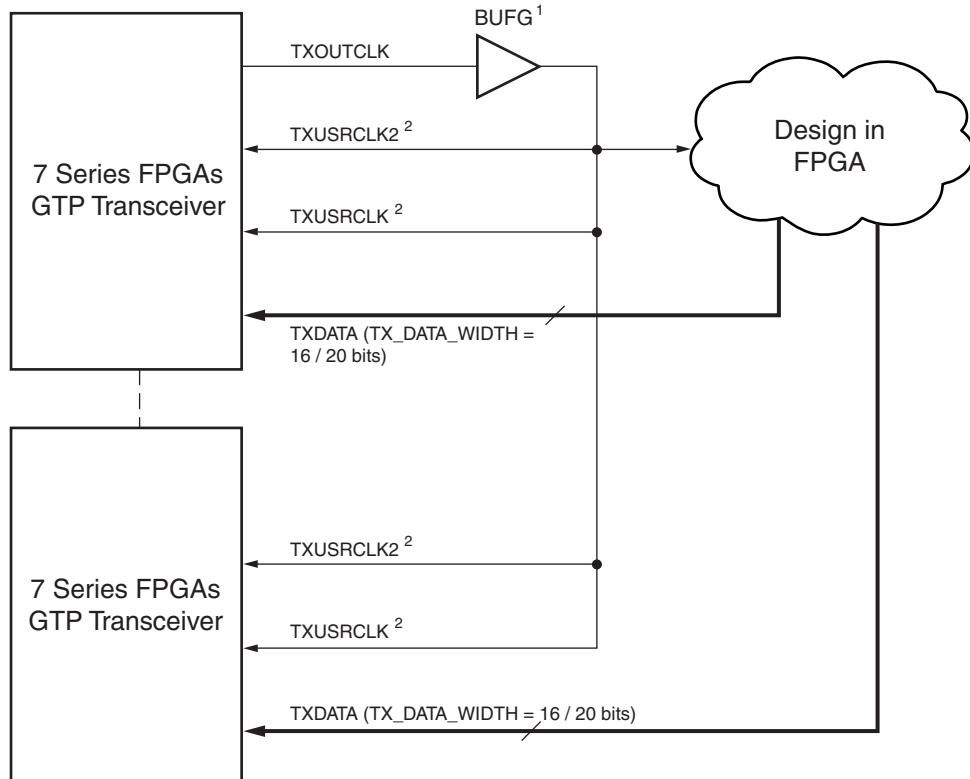
UG482\_c3\_02\_110911

*Figure 3-2: Single Lane—TXOUTCLK Drives TXUSRCLK2 (2-Byte Mode)*

Notes relevant to [Figure 3-2](#):

1. BUFH can be used with certain limitations. For details about placement constraints and restrictions on clocking resources (MMCM, BUFH, BUFG, etc.), refer to [UG472, 7 Series FPGAs Clocking Resources User Guide](#).
2.  $F_{TXUSRCLK2} = F_{TXUSRCLK}$ .

Similarly, Figure 3-3 shows the shows the same settings in multiple lanes configuration.



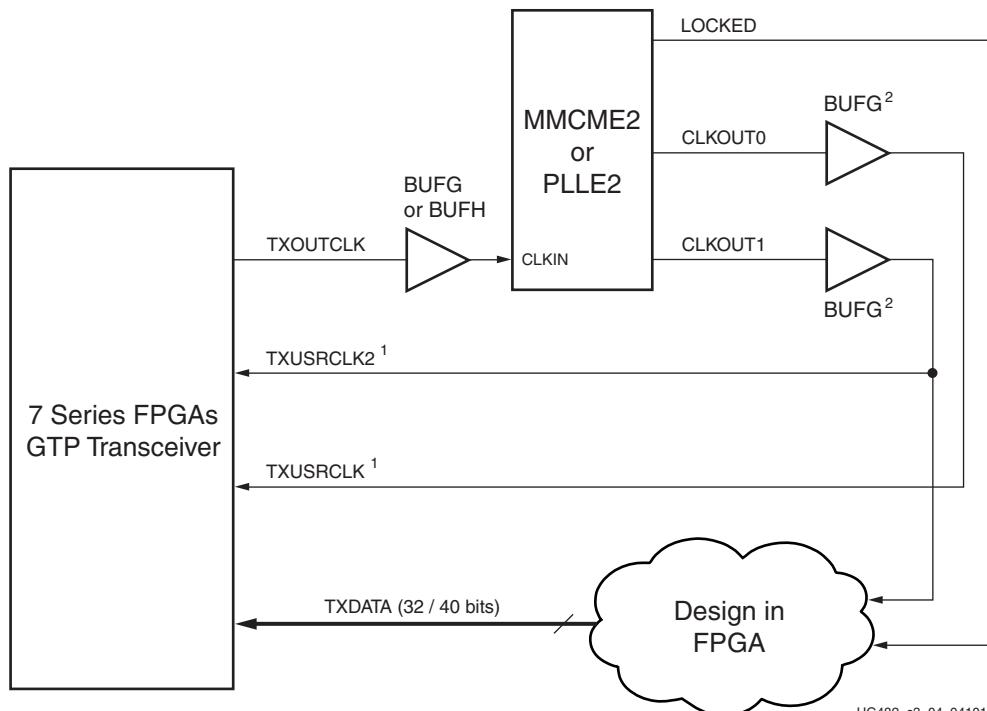
**Figure 3-3: Multiple Lanes—TXOUTCLK Drives TXUSRCLK2 (2-Byte Mode)**

Notes relevant to Figure 3-3:

1. BUFH can be used with certain limitations. For details about placement constraints and restrictions on clocking resources (MMCM, BUFH, BUFG, etc.), refer to [UG472, 7 Series FPGAs Clocking Resources User Guide](#).
2.  $F_{TXUSRCLK2} = F_{TXUSRCLK}$ .

## TXOUTCLK Driving GTP Transceiver TX in 4-Byte Mode

In [Figure 3-4](#), TXOUTCLK is used to drive TXUSRCLK2 for 4-byte mode ( $\text{TX\_DATA\_WIDTH} = 32$  or  $40$ ). The frequency of TXUSRCLK2 is equal to half of the frequency of TXUSRCLK. MMCMs or PLLs, which are part of the clock management tiles (CMTs) located in the top half of the device, can only drive the BUFGs in the top half of the devices. Similarly, MMCMs or PLLs located in the bottom half can only drive BUFGs in the bottom half.

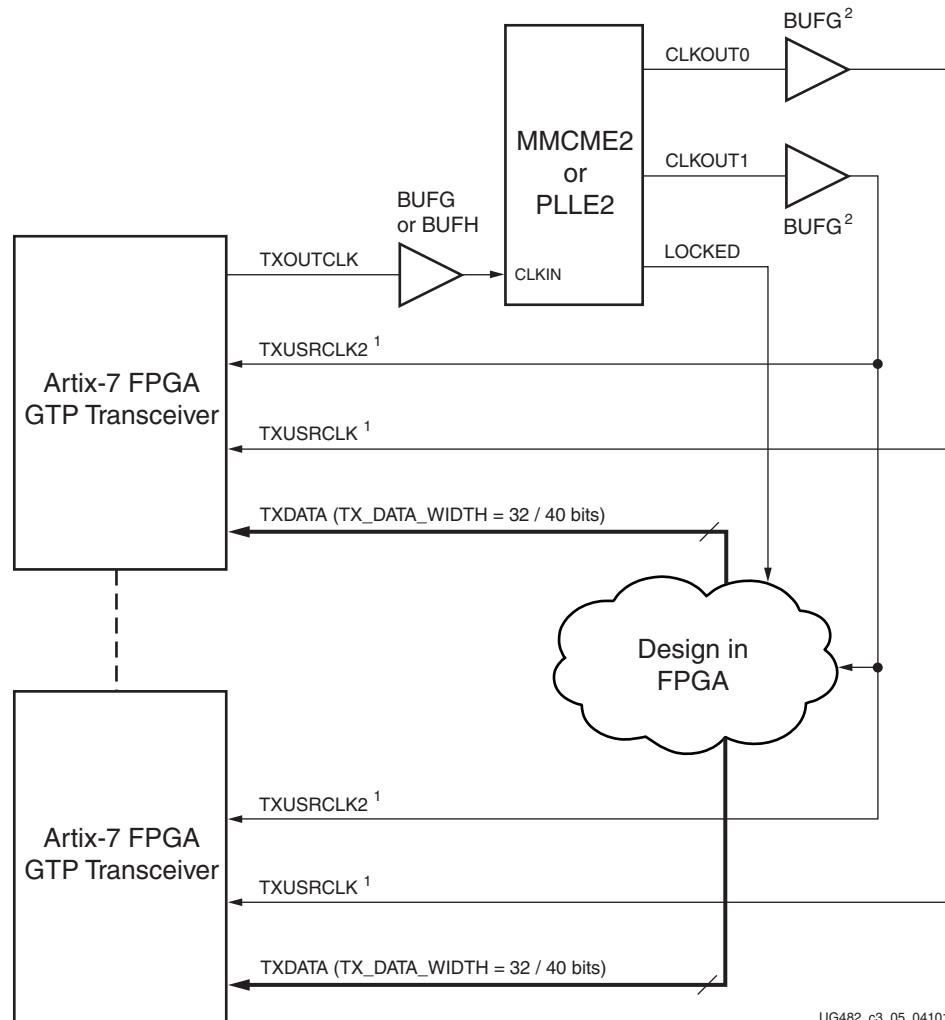


*Figure 3-4: Single Lane—TXOUTCLK Drives TXUSRCLK2 (4-Byte Mode)*

Notes relevant to [Figure 3-4](#):

1.  $F_{\text{TXUSRCLK2}} = F_{\text{TXUSRCLK}}/2$
2. In the XC7A200T device, BUFH can be used with certain limitations. For details about placement constraints and restrictions on clocking resources (MMCM, BUFH, BUFG, etc.), refer to [UG472, 7 Series FPGAs Clocking Resources User Guide](#).

Similarly, Figure 3-5 shows the shows the same settings in multiple lanes configuration.



UG482\_c3\_05\_041012

**Figure 3-5: Multiple Lanes—TXOUTCLK Drives TXUSRCLK2 (4-Byte Mode)**

Notes relevant to Figure 3-5:

1.  $F_{TXUSRCLK2} = F_{TXUSRCLK}/2$ .
2. In the XC7A200T device, BUFH can be used with certain limitations. For details about placement constraints and restrictions on clocking resources (MMCM, BUFH, BUFG, etc.), refer to [UG472, 7 Series FPGAs Clocking Resources User Guide](#).

# TX 8B/10B Encoder

## Functional Description

Many protocols use 8B/10B encoding on outgoing data. 8B/10B is an industry standard encoding scheme that trades two bits overhead per byte for achieved DC-balance and bounded disparity to allow reasonable clock recovery. The GTP transceiver has a built-in 8B/10B TX path to encode TX data without consuming FPGA resources. Enabling the 8B/10B encoder increases latency through the TX path. The 8B/10B encoder can be disabled or bypassed to minimize latency, if not needed.

### 8B/10B Bit and Byte Ordering

The order of the bits after the 8B/10B encoder is the opposite of the order shown in [Appendix C, 8B/10B Valid Characters](#), because 8B/10B encoding requires bit a0 to be transmitted first, and the GTP transceiver always transmits the right-most bit first. To match with 8B/10B, the 8B/10B encoder in the GTP transceiver automatically reverses the bit order. [Figure 3-6](#) shows data transmitted by the GTP transceiver when TX\_DATA\_WIDTH = 20 and 40. The number of bits used by TXDATA and corresponding byte orders are determined by TX\_DATA\_WIDTH.

- Only use TXDATA[15:0] if TX\_DATA\_WIDTH = 20
- Use full TXDATA[31:0] if TX\_DATA\_WIDTH = 40

When the 8B/10B encoder is bypassed and TX\_DATA\_WIDTH is set to a multiple of 10, 10-bit characters are passed to TX data interface with this format:

- The corresponding TXCHARDISPMODE represents the 9th bit
- The corresponding TXCHARDISPVAL represents the 8th bit
- The corresponding TXDATA byte represents [7:0] bits

### K Characters

The 8B/10B table includes special characters (K characters) that are often used for control functions. TXCHARISK ports are used to indicate if data on TXDATA are K characters or regular data. The 8B/10B encoder checks received TXDATA byte to match any K character if corresponding TXCHARISK bit is driven High.

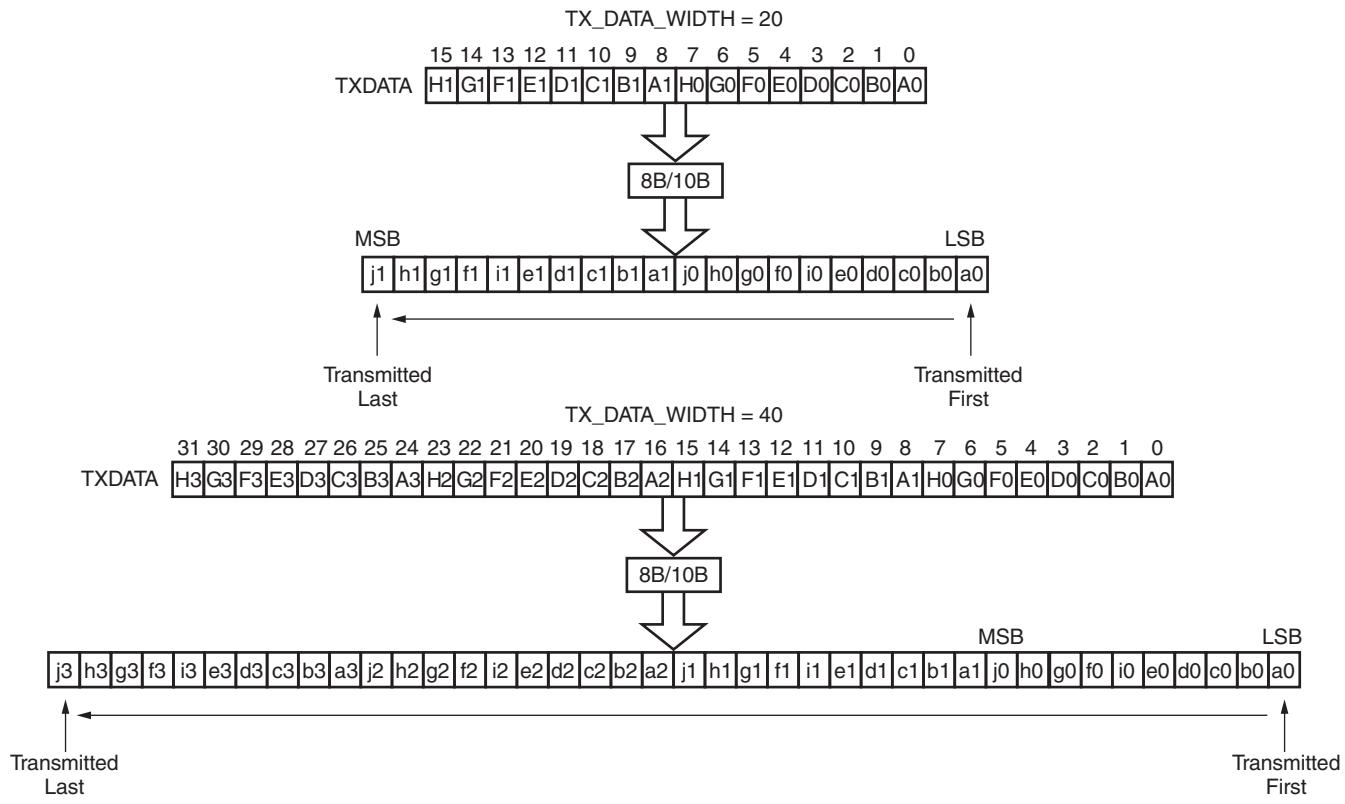


Figure 3-6: 8B/10B Bit and Byte Ordering

## Running Disparity

8B/10B coding is DC-balanced, meaning that the long-term ratio of 1s and 0s transmitted should be exactly 50%. To achieve this, the encoder always calculates the difference between the number of 1s transmitted and the number of 0s transmitted, and at the end of each character transmitted, makes the difference either +1 or -1. This difference is known as the *running disparity*.

To accommodate protocols that use disparity to send control information, the running disparity not only can be generated by the 8B/10B encoder but is also controllable through TXCHARDISPMODE and TXCHARDISPVAL as shown in Table 3-6. For example, an Idle character sent with reversed disparity might be used to trigger clock correction.

Table 3-6: TXCHARDISPMODE and TXCHARDISPVAL versus Outgoing Disparity

TXCHARDISPMODE	TXCHARDISPVAL	Outgoing Disparity
0	0	Calculated by the 8B/10B encoder.
0	1	Inverts running disparity when encoding TXDATA.
1	0	Forces running disparity negative when encoding TXDATA.
1	1	Forces running disparity positive when encoding TXDATA.

## Ports and Attributes

[Table 3-7](#) lists the ports required by the TX 8B/10B encoder.

**Note:** There are no TX encoder attributes.

**Table 3-7: TX 8B/10B Encoder Ports**

Port	Dir	Clock Domain	Description
TX8B10BBYPASS[3:0]	In	TXUSRCLK2	<p>This active-High port allows byte-interleaved data to bypass 8B/10B on a per-byte basis. TX8B10BEN must be High to use this per-byte bypass mode.</p> <p>TX8B10BBYPASS [3] corresponds to TXDATA[31:24]      TX8B10BBYPASS [2] corresponds to TXDATA[23:16]      TX8B10BBYPASS [1] corresponds to TXDATA[15:8]      TX8B10BBYPASS [0] corresponds to TXDATA[7:0]</p> <p>TX8B10BBYPASS[x] = 1, encoder for byte x is bypassed.      TX8B10BBYPASS[x] = 0, encoder for byte x is used.</p>
TX8B10BEN	In	TXUSRCLK2	<p>TX8B10BEN is set High to enable the 8B/10B encoder.      TX_DATA_WIDTH must be set to 20 or 40 when the 8B/10B encoder is enabled.</p> <p>0: 8B/10B encoder bypassed. This option reduces latency.      1: 8B/10B encoder enabled.</p>
TXCHARDISPMODE[3:0]	In	TXUSRCLK2	<p>Set High to work with TXCHARDISPVAL to force running disparity negative or positive when encoding TXDATA. Set Low to use normal running disparity. Refer to <a href="#">Table 3-6</a> for a detailed definition.</p> <p>TXCHARDISPMODE[3] corresponds to TXDATA[31:24]      TXCHARDISPMODE[2] corresponds to TXDATA[23:16]      TXCHARDISPMODE[1] corresponds to TXDATA[15:8]      TXCHARDISPMODE[0] corresponds to TXDATA[7:0]</p>
TXCHARDISPVAL[3:0]	In	TXUSRCLK2	<p>Work with TXCHARDISPMODE to provide running disparity control. Refer to <a href="#">Table 3-6</a> for detailed information.</p> <p>TXCHARDISPVAL[3] corresponds to TXDATA[31:24]      TXCHARDISPVAL[2] corresponds to TXDATA[23:16]      TXCHARDISPVAL[1] corresponds to TXDATA[15:8]      TXCHARDISPVAL[0] corresponds to TXDATA[7:0]</p>
TXCHARISK[3:0]	In	TXUSRCLK2	<p>When High, indicates the corresponding data byte on TXDATA is a valid K character.</p> <p>TXCHARISK[3] corresponds to TXDATA[31:24]      TXCHARISK[2] corresponds to TXDATA[23:16]      TXCHARISK[1] corresponds to TXDATA[15:8]      TXCHARISK[0] corresponds to TXDATA[7:0]</p> <p>A TXCHARISK bit should be driven Low when the corresponding data byte from TXDATA is set to bypass the 8B/10B encoder.</p>

## Enabling and Disabling 8B/10B Encoding

To enable the 8B/10B encoder, TX8B10BEN must be driven High. The TX 8B/10B encoder allows byte interleaved data to bypass the encoder on a per-byte basis. When TX8B10BEN is driven Low, all encoders are turned off and no data from TXDATA can be encoded.

When TX8B10BEN is High, driving a bit from TX8B10BBYPASS High can make the corresponding byte channel from TXDATA bypass 8B/10B encoding. When the encoder is turned off, the operation of the TXDATA port is as described in the FPGA TX interface.

## TX Gearbox

### Functional Description

Some high-speed data rate protocols use 64B/66B encoding to reduce the overhead of 8B/10B encoding while retaining the benefits of an encoding scheme. The TX gearbox provides support for 64B/66B and 64B/67B header and payload combining. The Interlaken interface protocol specification uses the 64B/67B encoding scheme. Refer to the Interlaken specification for further information. The Interlaken specification can be downloaded from: <http://www.interlakenalliance.com/>.

The TX gearbox supports 2-byte and 4-byte interfaces. Scrambling of the data is done in the FPGA logic.

### Ports and Attributes

*Table 3-8* defines the TX gearbox ports.

*Table 3-8: TX Gearbox Ports*

Port Name	Dir	Clock Domain	Description
TXGEARBOXREADY	Out	TXUSRCLK2	This output indicates if data can be applied to the 64B/66B or 64B/67B gearbox when GEARBOX_MODE is set to use the gearbox. 0: No data can be applied 1: Data must be applied
TXHEADER[2:0]	In	TXUSRCLK2	These ports are the header inputs. [1:0] are used for the 64B/66B gearbox, and [2:0] are used for the 64B/67B gearbox.
TXSEQUENCE[6:0]	In	TXUSRCLK2	These inputs are used for the fabric sequence counter when the TX gearbox is used. [5:0] are used for the 64B/66B gearbox, and [6:0] are used for the 64B/67B gearbox.
TXSTARTSEQ	In	TXUSRCLK2	This input indicates the first word to be applied after reset for the 64B/66B or 64B/67B gearbox.

[Table 3-9](#) defines the TX gearbox attributes.

**Table 3-9: TX Gearbox Attributes**

Attribute	Type	Description
GEARBOX_MODE	3-bit Binary	This attribute indicates the TX and RX gearbox modes: <ul style="list-style-type: none"><li>• Bit 2: Set to 0. Unused.</li><li>• Bit 1: Set to 0.<ul style="list-style-type: none"><li>0: Use the external sequence counter and apply inputs to TXSEQUENCE.</li><li>1: Not supported.</li></ul></li><li>• Bit 0:<ul style="list-style-type: none"><li>0: 64B/67B gearbox mode for Interlaken.</li><li>1: 64B/66B gearbox.</li></ul></li></ul>
TXGEARBOX_EN	String	When TRUE, this attribute enables the TX gearbox.

## Enabling the TX Gearbox

To enable the TX gearbox for the GTP transceiver, set the attribute TXGEARBOX\_EN to TRUE. The GEARBOX\_MODE attribute controls the GTP transceiver's TX and RX gearbox use modes.

## TX Gearbox Bit and Byte Ordering

[Figure 3-7](#) shows an example of the first four cycles of data entering and exiting the TX gearbox for 64B/66B encoding when using a 2-byte logic interface (TX\_DATA\_WIDTH = 16 (2-byte)). The input consists of a 2-bit header and 16 bits of data. On the first cycle, the header and 14 bits of data exit the TX gearbox. On the second cycle, the remaining two data bits from the previous cycle's TXDATA input along with 14 data bits from the current TXDATA input exit the TX gearbox. This continues for the third and fourth cycle. On the fifth cycle, the output of the TX gearbox contains two remaining data bits from the first 66-bit block, the header of the second 66-bit block, and 28 data bits from the second 66-bit block.

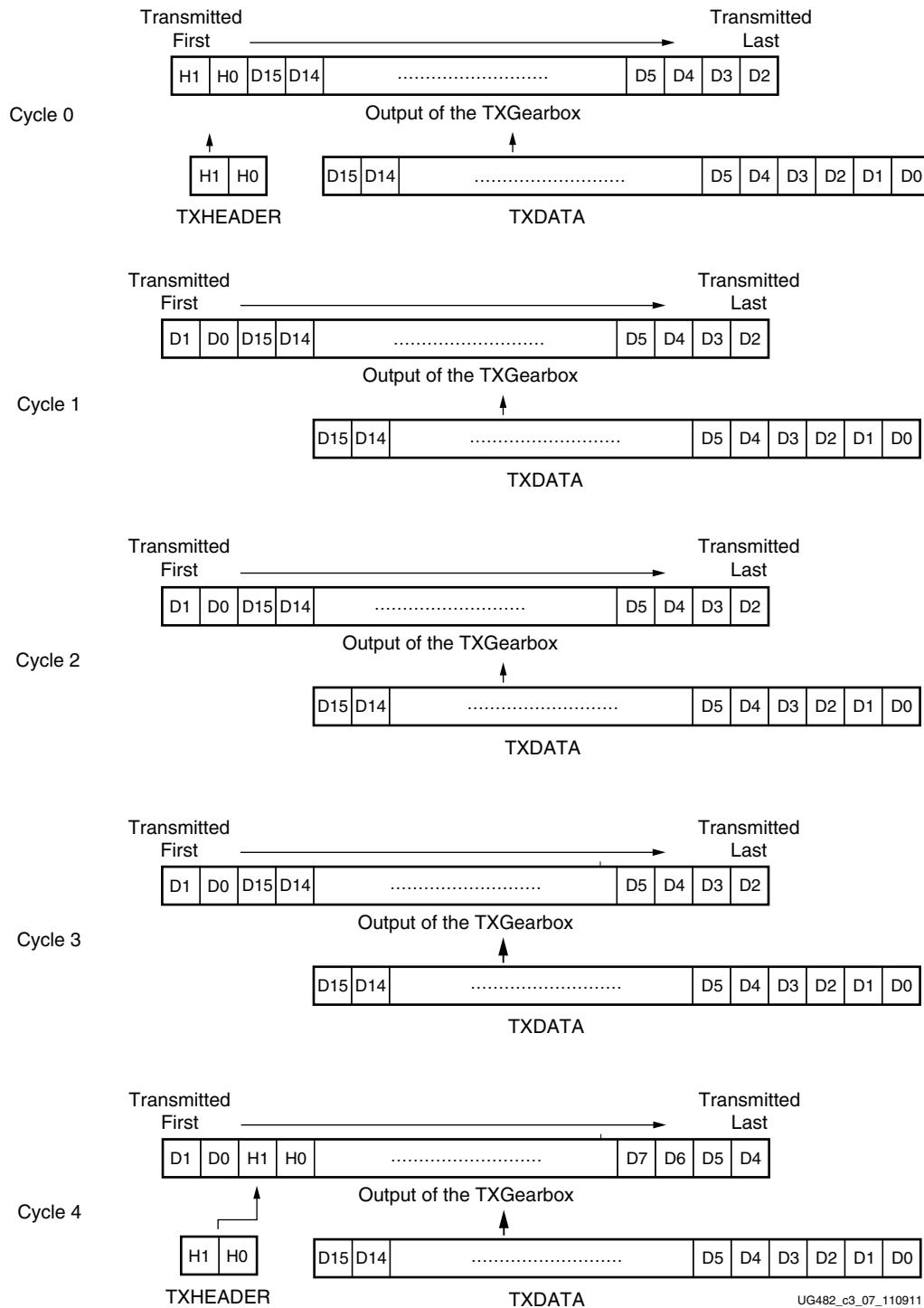


Figure 3-7: TX Gearbox Bit Ordering

Note relevant to [Figure 3-7](#):

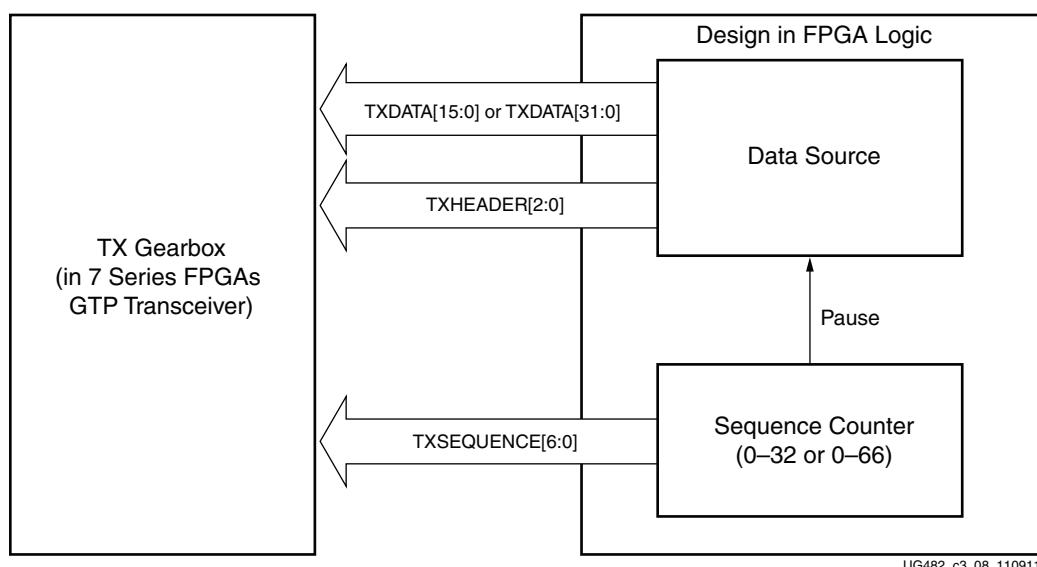
1. Per IEEE802.3ae nomenclature, H1 corresponds to TxB<0>, H0 to TxB<1>, etc.

## TX Gearbox Operating Modes

The GTP transceiver's TX gearbox only supports the external sequence counter mode and this must be implemented in user logic. The TX gearbox supports 2-byte and 4-byte interfaces to the FPGA logic.

### External Sequence Counter Operating Mode

As shown in [Figure 3-8](#), the external sequence counter operating mode uses the TXSEQUENCE [6:0], TXDATA[31:0], and TXHEADER[2:0] inputs. A binary counter must exist in the user logic to drive the TXSEQUENCE port. For 64B/66B encoding, the counter increments from 0 to 32 and repeats from 0. For 64B/67B encoding, the counter increments from 0 to 66 and repeats from 0. When using 64B/66B encoding, tie TXSEQUENCE [6] to logic 0 and tie the unused TXHEADER [2] to logic 0. The sequence counter increment ranges ({0 to 32}, {0 to 66}) are identical for 2-byte and 4-byte interfaces. However, the counter must increment once every two TXUSRCLK2 cycles when using a 2-byte interface and every TXUSRCLK2 cycle when using a 4-byte interface.



*Figure 3-8: TX Gearbox in External Sequence Counter Mode*

Due to the nature of the 64B/66B and 64B/67B encoding schemes, user data is held (paused) during various sequence counter values. Data is paused for two TXUSRCLK2 cycles in 2-byte mode and for one TXUSRCLK2 cycle in 4-byte mode. Valid data transfer is resumed on the next TXUSRCLK2 cycle. The data pause only applies to TXDATA and not to TXHEADER. The TXSEQUENCE pause locations for various modes are described in [Table 3-10](#) and [Table 3-11](#).

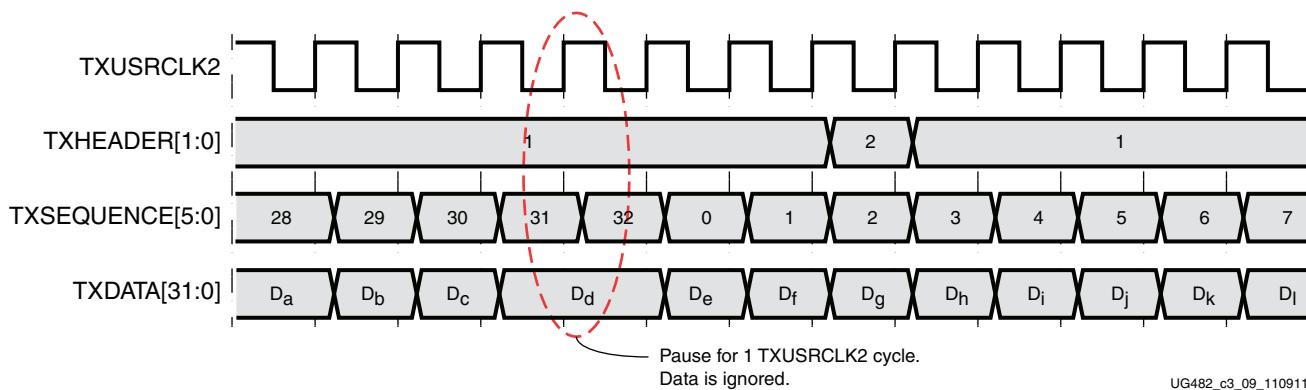
**Table 3-10: 64B/66B Encoding Frequency of TXSEQUENCE and Pause Locations**

TX_DATA_WIDTH	Frequency of TXSEQUENCE	TXSEQUENCE PAUSE
32 (4-byte)	1 X TXUSRCLK2	31
16 (2-byte)	2 X TXUSRCLK2	31

**Table 3-11: 64B/67B Encoding Frequency of TXSEQUENCE and Pause Locations**

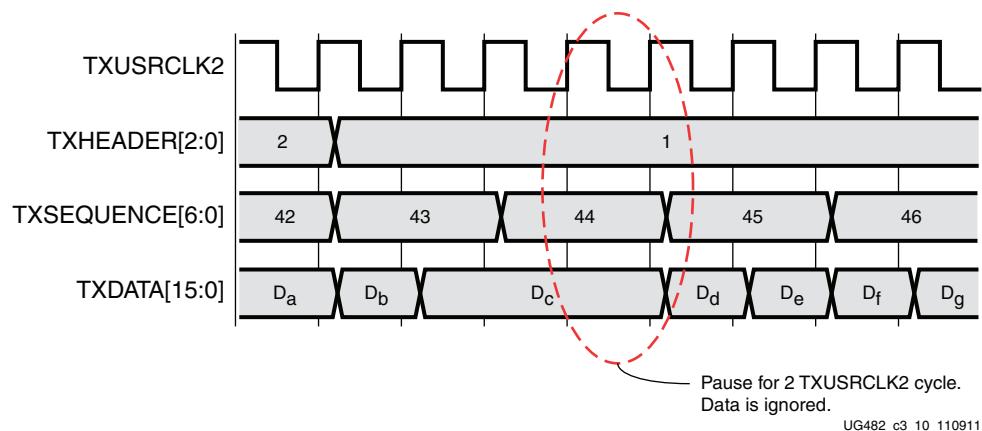
TX_DATA_WIDTH	Frequency of TXSEQUENCE	TXSEQUENCE PAUSE
32 (4-byte)	1 X TXUSRCLK2	21, 44, 65
16 (2-byte)	2 X TXUSRCLK2	21, 44, 65

[Figure 3-9](#) shows how a pause occurs at counter value 31 when using a 4-byte fabric interface in external sequence counter mode with 64B/66B encoding.



**Figure 3-9: Pause at Sequence Counter Value 31**

[Figure 3-10](#) shows how a pause occurs at counter value 44 when using a 2-byte fabric interface in external sequence counter mode with 64B/67B encoding.



**Figure 3-10: Pause at Sequence Counter Value 44**

The sequence of transmitting 64/67 data for the external sequence counter mode is:

1. Apply GTTXRESET and wait until the reset cycle is completed.
2. During reset, apply 7'h00 to TXSEQUENCE, header information to TXHEADER, and initial data to TXDATA. This state can be held indefinitely until data transmission is ready.
3. On count 0, apply data to TXDATA and header information to TXHEADER. For a 2-byte interface (TX\_DATA\_WIDTH = 16), drive the second 2 bytes to TXDATA while still on count 0.
4. The sequence counter increments to 1 while data is driven on TXDATA.
5. After applying 4 bytes of data, the counter increments to 2. Apply data on TXDATA and header information on TXHEADER.
6. On count 21, stop data pipeline.
7. On count 22, drive data on TXDATA.
8. On count 44, stop data pipeline.

9. On count 45, drive data on TXDATA.
10. On count 65, stop data pipeline.
11. On count 66, drive data on TXDATA.

The sequence of transmitting 64/66 data for the external sequence counter mode is as follows:

1. Apply GTTXRESET and wait until the reset cycle is completed.
2. During reset, apply 6'h00 to TXSEQUENCE, the appropriate header data to TXHEADER, and initial data to TXDATA. This state can be held indefinitely until data transmission is ready.
3. On count 0, apply data to TXDATA and header information to TXHEADER. For a 2-byte interface (TX\_DATA\_WIDTH = 16), drive the second 2 bytes to TXDATA while still on count 0.
4. The sequence counter increments to 1 while data is driven on TXDATA.
5. After applying 4 bytes of data, the counter increments to 2. Drive data on TXDATA and header information on TXHEADER.
6. On count 31, stop data pipeline.
7. On count 32, drive data on TXDATA.

# TX Buffer

## Functional Description

The GTP transceiver TX datapath has two internal parallel clock domains used in the PCS: the PMA parallel clock domain (XCLK) and the TXUSRCLK domain. To transmit data, the XCLK rate must match the TXUSRCLK rate, and all phase differences between the two domains must be resolved. [Figure 3-11](#) shows the XCLK and TXUSRCLK domains.

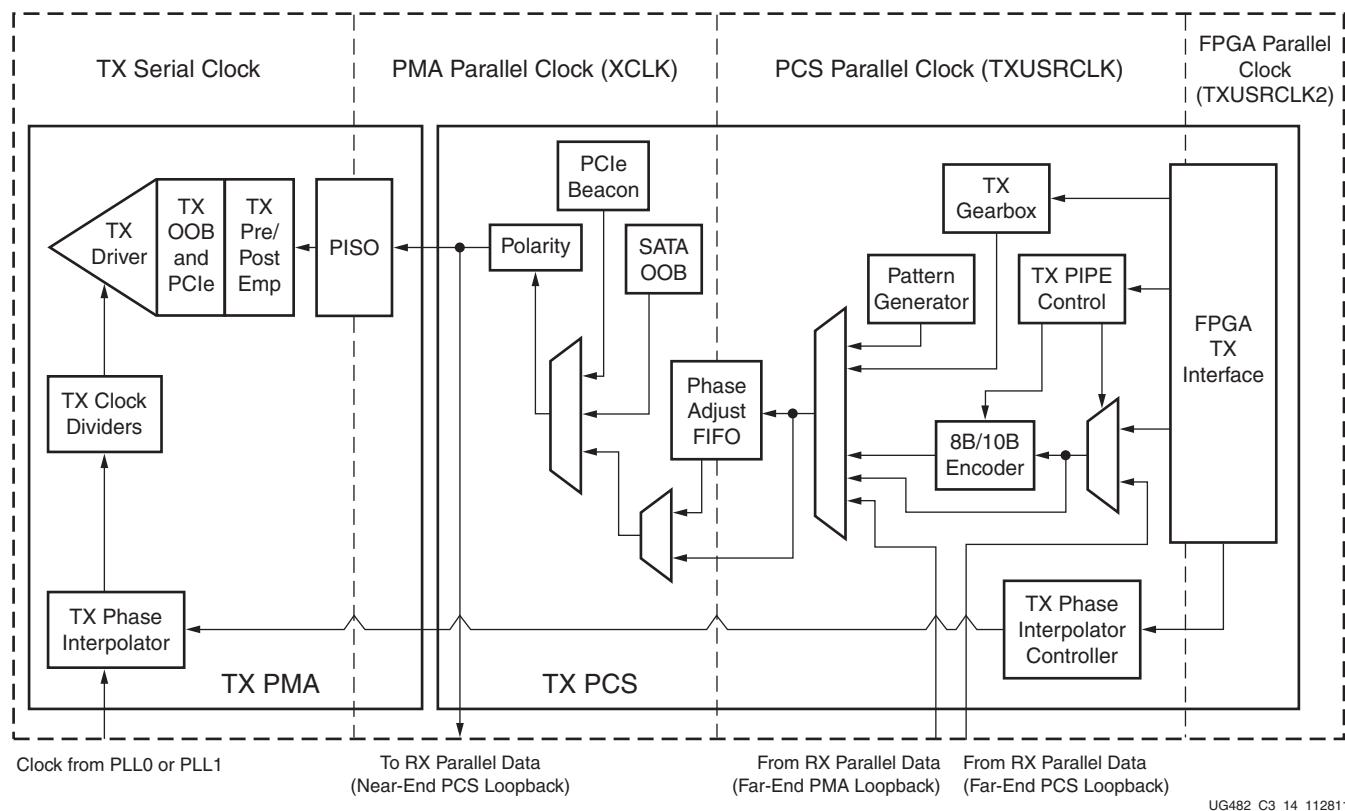


Figure 3-11: TX Clock Domains

The GTP transceiver transmitter includes a TX buffer and a TX phase alignment circuit to resolve phase differences between the XCLK and TXUSRCLK domains. The TX phase alignment circuit is used when TX buffer is bypassed (see [TX Pattern Generator, page 105](#)). All TX datapaths must use either the TX buffer or the TX phase alignment circuit. [Table 3-12](#) shows trade-offs between buffering and phase alignment.

**Table 3-12: TX Buffering versus Phase Alignment**

	<b>TX Buffer</b>	<b>TX Phase Alignment</b>
Ease of Use	The TX buffer is the recommended default to use when possible. It is robust and easier to operate.	Phase alignment is an advanced feature that requires extra logic and additional constraints on clock sources. TXOUTCLKSEL must select the GTP transceiver reference clock as the source of TXOUTCLK to drive TXUSRCLK.
Latency	If low latency is critical, the TX buffer must be bypassed.	Phase alignment uses fewer registers in the TX datapath to achieve lower and deterministic latency.
TX Lane-to-Lane Deskew		The TX phase alignment circuit can be used to reduce the lane skew between separate GTP transceivers. All GTP transceivers involved must use the same line rate.

## Ports and Attributes

[Table 3-13](#) defines the TX buffer ports.

**Table 3-13: TX Buffer Ports**

<b>Port</b>	<b>Dir</b>	<b>Clock Domain</b>	<b>Description</b>
TXBUFSTATUS[1:0]	Out	TXUSRCLK2	TX buffer status. TXBUFSTATUS[1]: TX buffer overflow or underflow status. When TXBUFSTATUS[1] is set High, it remains High until the TX buffer is reset. 1: TX FIFO has overflow or underflow. 0: No TX FIFO overflow or underflow error. TXBUFSTATUS[0]: TX buffer fullness. 1: TX FIFO is at least half full. 0: TX FIFO is less than half full.

[Table 3-14](#) defines the TX buffer attributes.

**Table 3-14: TX Buffer Attributes**

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
TXBUF_EN	String	Use or bypass the TX buffer. TRUE: Uses the TX buffer (default). FALSE: Bypasses the TX buffer (advanced feature).

Table 3-14: TX Buffer Attributes (Cont'd)

Attribute	Type	Description
TX_XCLK_SEL	String	Selects the clock source used to drive the PMA parallel clock domain (XCLK).  TXOUT: Selects TXOUTCLK as source of XCLK. Use when using the TX buffer.  TXUSR: Selects TXUSRCLK as source of XCLK. Used when bypassing the TX buffer.
TXBUF_RESET_ON_RATE_CHANGE	String	GTP transceiver internally generated TX buffer reset on rate change.  TRUE: Enables auto TX buffer reset on rate change.  FALSE: Disables auto TX buffer reset on rate change.

## Using the TX Buffer

The TX buffer should be reset whenever TXBUFSTATUS indicates an overflow or underflow condition. The TX buffer can be reset by using GTTXRESET, TXPCSRESET, or the GTP transceiver internally generated TX buffer reset on rate change when TXBUF\_RESET\_ON\_RATE\_CHANGE = TRUE. Assertion of GTTXRESET triggers a sequence that resets the entire transmitter of the GTP transceiver. These settings are used to enable the TX buffer to resolve phase differences between the XCLK and TXUSRCLK domains:

- TXBUF\_EN = TRUE  
TX\_XCLK\_SEL = TXOUT

## TX Buffer Bypass

### Functional Description

Bypassing the TX buffer is an advanced feature of the 7 series GTP transceiver. The TX phase-alignment circuit is used to adjust the phase difference between the PISO parallel clock domain and the TX XCLK domain to transfer data from the PCS into the PISO. It also performs the TX delay alignment by continuously adjusting the TXUSRCLK to compensate for temperature and voltage variations. The combined TX phase and delay alignments must be manually controlled by the user. [Figure 3-11, page 95](#) shows the XCLK and TXUSRCLK domains. [Table 3-12, page 96](#) describes the trade-offs between buffering and phase alignment.

### Ports and Attributes

[Table 3-15](#) defines the TX buffer bypass ports.

Table 3-15: TX Buffer Bypass Ports

Port	Dir	Clock Domain	Description
TXPHDLYRESET	In	Async	TX phase alignment hard reset. Forces TXOUTCLK to the center of the delay alignment tap. The delay alignment tap has a full range of $\pm 4$ ns and a half range of $\pm 2$ ns. The user is recommended to use TXDLYSRESET only for phase and delay alignment.
TXPHALIGN	In	Async	Sets the TX phase alignment.
TXPHALIGNEN	In	Async	Enables the TX phase alignment.
TXPHDLYPD	In	Async	TX phase and delay alignment circuit power down. TXPHDLYPD is tied High when: <ul style="list-style-type: none"> <li>TX buffer bypass is not in use.</li> <li>TXPD is asserted.</li> <li>TXOUTCLKSEL is set to 3'b011 or 3'b100 but the reference clock is not connected.</li> </ul> TXPHDLYPD is tied Low during TX buffer bypass mode normal operation. 0: Power-up the TX phase and delay alignment circuit. 1: Power-down the TX phase and delay alignment circuit.
TXPHINIT	In	Async	TX phase alignment initialization.
TXPHOVRDEN	In	Async	TX phase alignment counter override enable. Tied Low when not in use. 0: Normal operation. 1: Enables TX phase alignment counter override with the value from TXPH_CFG[10:0].
TXDLYSRESET	In	Async	TX delay alignment soft reset to gradually shift TXOUTCLK to the center of the delay alignment tap. The delay alignment tap has a full range of $\pm 4$ ns and a half range of $\pm 2$ ns. TXPHDLYRESET and GTTXRESET force TXOUTCLK to the center of the delay alignment tap which might cause a sudden phase shift within one clock cycle. Use TXPMARESET followed by TXDLYSRESET to reset the transmitter and restart phase alignment without sudden phase shifts on TXOUTCLK.
TXDLYBYPASS	In	Async	TX delay alignment bypass. 0: Uses the TX delay alignment circuit. 1: Bypasses the TX delay alignment circuit.
TXDLYEN	In	Async	Enables the TX delay alignment.
TXDLYOVRDEN	In	Async	TX delay alignment counter override enable. Tied Low when not in use. 0: Normal operation. 1: Enables TX delay alignment counter override with the value from TXDLY_CFG[14:6].
TXPHDLYTSTCLK	In	Async	TX phase and delay alignment test clock. Used with TXDLYHOLD and TXDLYUPDOWN.
TXDLYHOLD	In	TXPHDLYTSTCLK	TX delay alignment hold. Used as a hold override when TXPHDLY_CFG[1] = 1 to bypass the TX phase and delay alignment voter. Tied Low when not in use.
TXDLYUPDOWN	In	TXPHDLYTSTCLK	TX delay alignment up or down. Used as an up or down override when TXPHDLY_CFG[1] = 1 to bypass the TX phase and delay alignment voter. Tied Low when not in use.
TXPHALIGNDONE	Out	Async	TX phase alignment done.

Table 3-15: TX Buffer Bypass Ports (Cont'd)

Port	Dir	Clock Domain	Description
TXPHINITDONE	Out	Async	Indicates that TX phase alignment initialization is done.
TXDLYSRESETDONE	Out	Async	Indicates that TX delay alignment soft reset is done.
TXSYNCMODE	In	Async	Reserved. Tie to GND.
TXSYNCCALLIN	In	Async	Reserved. Tie to GND.
TXSYNCIN	In	Async	Reserved. Tie to GND.
TXSYNCOUT	Out	Async	Reserved.
TXSYNCDONE	Out	Async	Reserved.

Table 3-16: TX Buffer Attributes

Attribute	Type	Description
TXBUF_EN	String	Use or bypass the TX buffer. TRUE: Uses the TX buffer (default). FALSE: Bypasses the TX buffer (advanced feature).
TX_XCLK_SEL	String	Selects the clock source used to drive the PMA parallel clock domain (XCLK). TXOUT: Selects TXOUTCLK as the source of XCLK. Used when using the TX buffer. TXUSR: Selects TXUSRCLK as the source of XCLK. Used when bypassing the TX buffer.
TXPH_CFG	16-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TXPH_MONITOR_SEL	5-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TXPHDLY_CFG	24-bit Binary	TX phase and delay alignment configuration. TXPHDLY_CFG[19] = 1 is used to set the TX delay alignment tap to the full range of $\pm 4$ ns. TXPHDLY_CFG[19] = 0 is used to set the TX delay alignment tap to the half range of $\pm 2$ ns. Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TXDLY_CFG	16-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TXDLY_LCFG	9-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TXDLY_TAP_CFG	16-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TXSYNC_MULTILANE	1-bit Binary	Reserved. Tie to 1'b0.
TXSYNC_SKIP_DA	1-bit Binary	Reserved. Tie to 1'b0.

Table 3-16: TX Buffer Attributes (Cont'd)

Attribute	Type	Description
TXSYNC_OVRD	1-bit Binary	Reserved. Tie to 1'b1.
LOOPBACK_CFG	1-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

## TX Buffer Bypass Use Modes

TX phase alignment can be performed on one channel (single lane) or a group of channels sharing a single TXOUTCLK (multi-lane). For GTP transceivers, TX buffer bypass supports single-lane and multi-lane applications (see [Table 3-17](#)).

Table 3-17: TX Buffer Bypass Use Modes

TX Buffer Bypass
Single Lane
Multi-lane

### Using TX Buffer Bypass in Single-Lane Mode

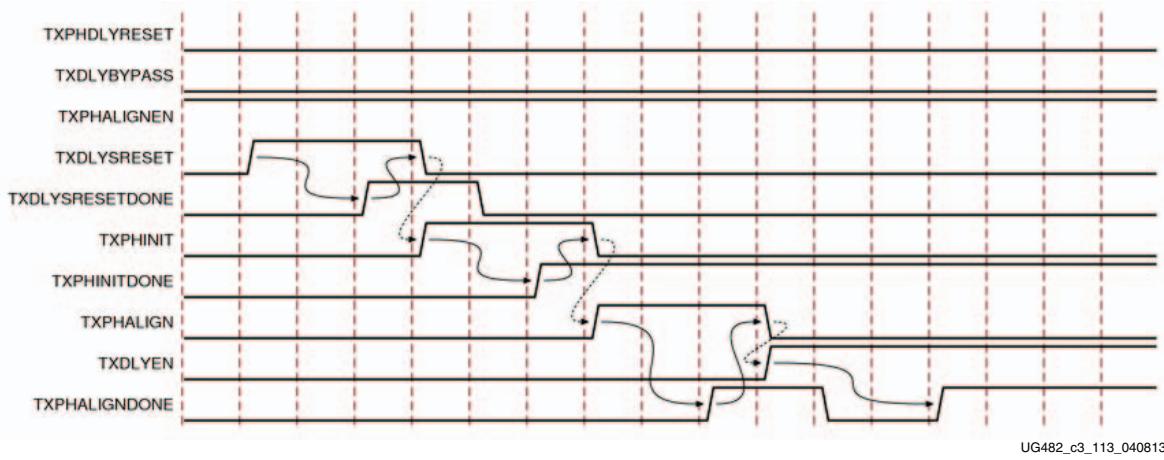
These GTP transceiver settings should be used to bypass the TX buffer:

- TXBUF\_EN = FALSE
- TX\_XCLK\_SEL = TXUSR
- TXOUTCLKSEL = 3'b011 or 3'b100 to select the GTP transceiver reference clock as the source of TXOUTCLK

With the GTP transceiver reference clock selected, TXOUTCLK is used as the source of the TXUSRCLK. The user must ensure that TXOUTCLK and the selected GTP transceiver reference clock are operating at the desired frequency. When the TX buffer is bypassed, the TX phase alignment procedure must be performed after these conditions:

- Resetting or powering up the GTP transceiver TX
- Resetting or powering up the PLL
- Change of the GTP transceiver reference clock source or frequency
- Change of the TX line rate

[Figure 3-12](#) shows the required steps to perform the TX phase alignment and use the TX delay alignment to adjust TXUSRCLK to compensate for temperature and voltage variations.



**Figure 3-12: TX Buffer Bypass Example, Single Lane Mode**

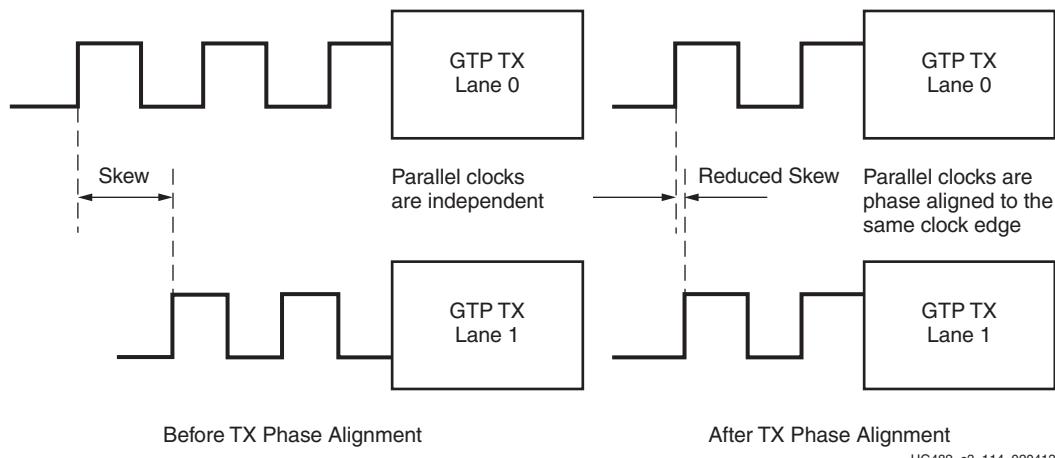
Notes relevant to [Figure 3-12](#):

1. The sequence of events in [Figure 3-12](#) is not drawn to scale.
2. Set the TXSYNC\_OVRD attribute to 1'b1.
3. Set TXPHDLYRESET and TXDLYBYPASS to Low for all lanes.
4. Set TXPHALIGNEN to High.
5. Assert TXDLYSRESET. Hold this signal High until TXDLYSRESETDONE is asserted.
6. Deassert TXDLYSRESET after TXDLYSRESETDONE is asserted.
7. When TXDLYSRESET is deasserted, assert TXPHINIT. Hold this signal High until the rising edge of TXPHINITDONE is observed.
8. Deassert TXPHINIT.
9. Assert TXPHALIGN. Hold this signal High until the rising edge of TXPHALIGNDONE is observed.
10. Deassert TXPHALIGN.
11. Assert TXDLYEN. This causes TXPHALIGNDONE to be deasserted.
12. Hold TXDLYEN until the rising edge of TXPHALIGNDONE is observed.
13. TX delay alignment continues to adjust TXUSRCLK to compensate for temperature and voltage variations.

## Using the TX Phase Alignment to Minimize the TX Lane-to-Lane Skew

The TX phase alignment circuit can also be used to minimize skew between GTP transceivers. [Figure 3-13](#) shows how the TX phase alignment circuit can reduce lane skew by aligning the XCLK domains of multiple GTP transceivers to a common clock source. Figure 3-22 shows multiple GTP transceiver lanes running before and after TX phase is aligned to a common clock. Before the TX phase alignment, all XCLKs have an arbitrary phase difference. After TX phase alignment, the only phase difference is the skew from the common clock, and all lanes transmit data simultaneously as long as the datapath latency is matched. TXUSRCLK and TXUSRCLK2 for all GTP transceivers must come from the

same source and must be routed through a low skew clocking resource such as a BUFG for the TX phase alignment circuit to be effective.



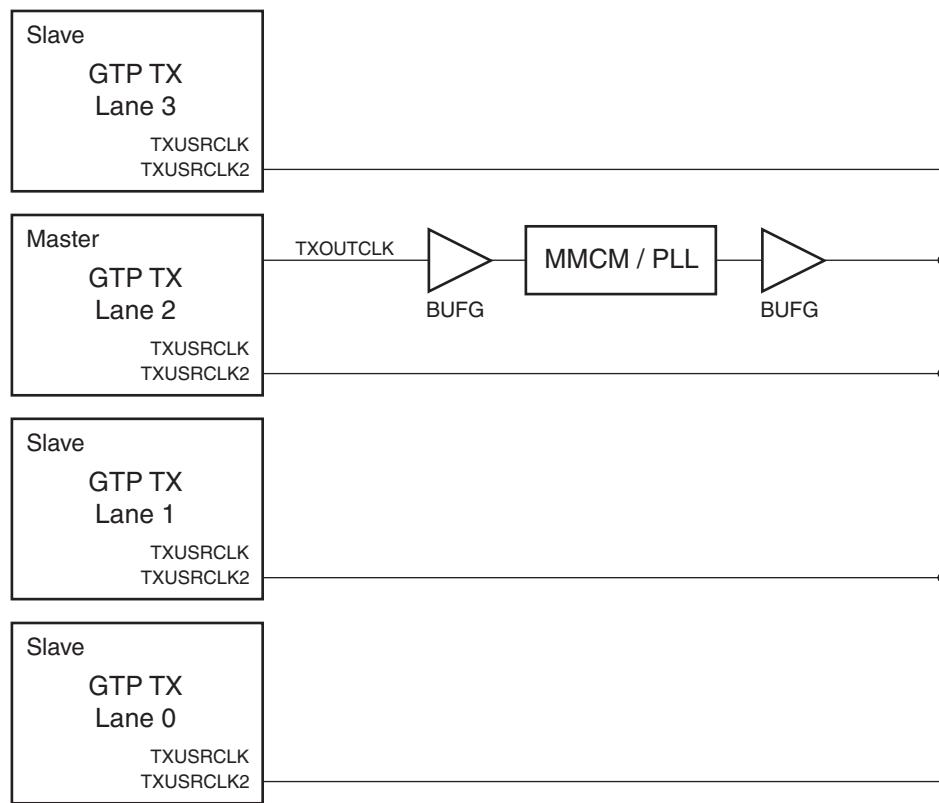
**Figure 3-13: TX Phase Alignment to Minimize TX Lane-to-Lane Skew**

### Using TX Buffer Bypass in Multi-Lane Mode

This section describes the steps required to perform the multi-lane TX buffer bypass alignment procedure.

- **Master:** In a multi-lane application, the buffer bypass master is the lane that is the source of TXOUTCLK.
- **Slave:** All the lanes that share the same TXUSRCLK/TXUSRCLK2, which is generated from the TXOUTCLK of the buffer bypass master.

Figure 3-14 shows an example of buffer bypass master versus slave lanes.



UG482\_c3\_115\_020413

**Figure 3-14: TX Buffer Bypass in Multi-Lane Mode Example**

These GTP transceiver settings are used to bypass the TX buffer:

- TXBUF\_EN = FALSE
- TX\_XCLK\_SEL = TXUSR
- TXOUTCLKSEL = 3'b011 or 3'b100 to select the GTP transceiver reference clock as the source of TXOUTCLK

With the GTP transceiver reference clock selected, TXOUTCLK is used as the source of the TXUSRCLK. The user must ensure that TXOUTCLK and the selected GTP transceiver reference clock is running and operating at the desired frequency. When the TX buffer is bypassed, the TX phase alignment procedure must be performed after these events:

- Resetting or powering up the GTP transceiver transmitter
- Resetting or powering up the PLL
- Change of the GTP transceiver reference clock source or frequency
- Change of the TX line rate

Figure 3-15 shows the required steps to perform TX phase and delay alignment.

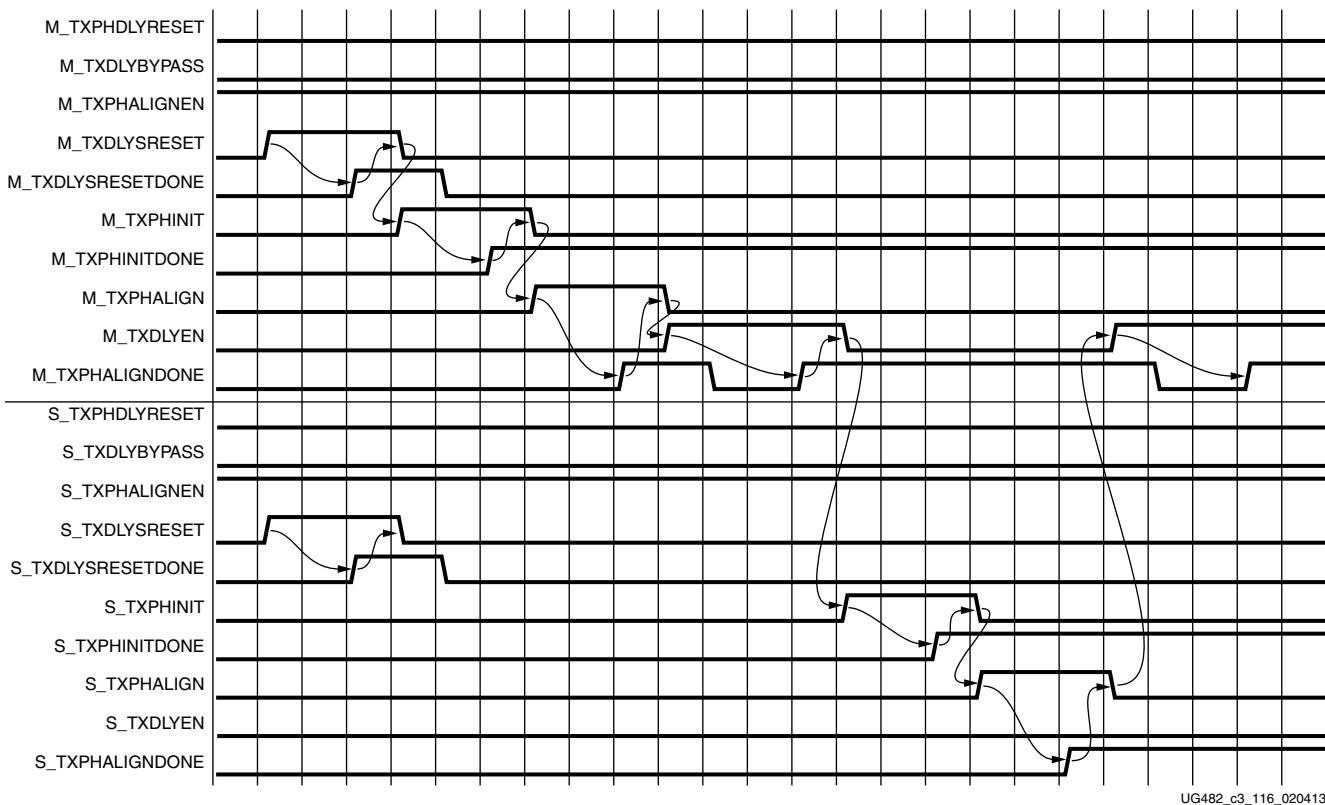


Figure 3-15: TX Phase and Delay Alignment, Multi-Lane Mode

Notes relevant to Figure 3-15:

1. The sequence of events shown in Figure 3-15 is not drawn to scale.
2. M\_\* denotes ports related to the master lane.
3. S\_\* denotes ports related to the slave lane(s).
4. GTP transceiver: Set the TXSYNC\_OVRD attribute to 1'b1.
5. Set TXPHDLYRESET and TXDLYBYPASS to Low for all lanes.
6. Set TXPHALIGNEN to High for all lanes.
7. Assert TXDLYSRESET for all lanes. Hold this signal High until TXDLYSRESETDONE of the respective lane is asserted.
8. Deassert TXDLYSRESET for the lane in which the TXDLYSRESETDONE is asserted.
9. When TXDLYSRESET of all lanes are deasserted, assert TXPHINIT for the master lane. Hold this signal High until the rising edge of TXPHINITDONE of the master lane is observed.
10. Deassert TXPHINIT for the master lane.
11. Assert TXPHALIGN for the master lane. Hold this signal High until the rising edge of TXPHALIGNDONE of the master lane is observed.
12. Deassert TXPHALIGN for the master lane.
13. Assert TXDLYEN for the master lane. This causes TXPHALIGNDONE to be deasserted.

14. Hold TXDLYEN for the master lane High until the rising edge of TXPHALIGNDONE of the master lane is observed.
15. Deassert TXDLYEN for the master lane.
16. Assert TXPHINIT for all slave lane(s). Hold this signal High until the rising edge of TXPHINITDONE of the respective slave lane is observed.
17. Deassert TXPHINIT for the slave lane in which the TXPHINITDONE is asserted.
18. When TXPHINIT for all slave lane(s) are deasserted, assert TXPHALIGN for all slave lane(s). Hold this signal High until the rising edge of TXPHALIGNDONE of the respective slave lane is observed.
19. Deassert TXPHALIGN for the slave lane in which the TXPHALIGNDONE is asserted.
20. When TXPHALIGN for all slave lane(s) are deasserted, assert TXDLYEN for the master lane. This causes TXPHALIGNDONE of the master lane to be deasserted.
21. Wait until TXPHALIGNDONE of the master lane reasserts. Phase and delay alignment for the multi-lane interface is complete. Continue to hold TXDLYEN for the master lane High to adjust TXUSRCLK to compensate for temperature and voltage variations.

## TX Pattern Generator

### Functional Description

Pseudo-random bit sequences (PRBS) are commonly used to test the signal integrity of high-speed links. These sequences appear random but have specific properties that can be used to measure the quality of a link. The GTP transceiver pattern generator block can generate several industry-standard PRBS patterns listed in [Table 3-18](#).

*Table 3-18: Supported PRBS Patterns*

Name	Polynomial	Length of Sequence	Description
PRBS-7	$1 + X^6 + X^7$	$2^7 - 1$ bits	Used to test channels with 8B/10B.
PRBS-15	$1 + X^{14} + X^{15}$	$2^{15} - 1$ bits	ITU-T Recommendation O.150, Section 5.3. PRBS-15 is often used for jitter measurement because it is the longest pattern the Agilent DCA-J sampling scope can handle.
PRBS-23	$1 + X^{18} + X^{23}$	$2^{23} - 1$ bits	ITU-T Recommendation O.150, Section 5.6. PRBS-23 is often used for non-8B/10B encoding schemes. It is one of the recommended test patterns in the SONET specification.
PRBS-31	$1 + X^{28} + X^{31}$	$2^{31} - 1$ bits	ITU-T Recommendation O.150, Section 5.8. PRBS-31 is often used for non-8B/10B encoding schemes. It is a recommended PRBS test pattern for 10 Gigabit Ethernet. See IEEE 802.3ae-2002.

In addition to PRBS patterns, the GTP transceiver supports 16-UI or 20-UI square wave test patterns, depending on data width as well as a 2-UI square wave test pattern and PCI Express compliance pattern generation (see [Table 3-19](#) and [Figure 3-16](#)). Clocking patterns are usually used to check PLL random jitter often done with a spectrum analyzer.

Table 3-19: PCI Express Compliance Pattern

Symbol	K28.5	D21.5	K28.5	D10.2
Disparity	0	1	1	0
Pattern	0011111010	1010101010	1100000101	0101010101

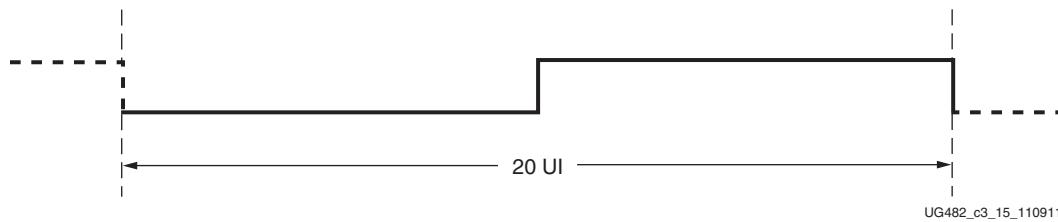


Figure 3-16: 20-UI Square Wave

The error insertion function is supported to verify link connection and also for jitter tolerance tests. When an inverted PRBS pattern is necessary, the TXPOLARITY signal is used to control polarity. [Figure 3-17](#) shows the TX pattern generator block.

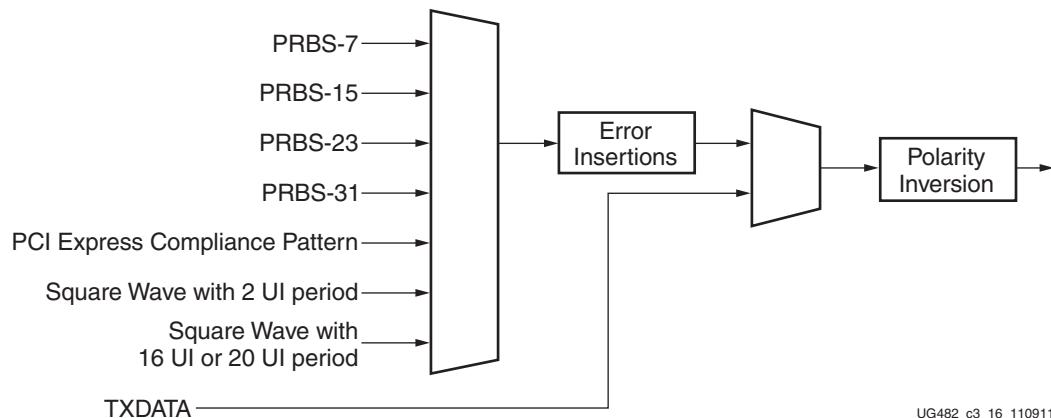


Figure 3-17: TX Pattern Generator Block

## Ports and Attributes

[Table 3-20](#) defines the pattern generator ports.

**Table 3-20: Pattern Generator Ports**

<b>Port Name</b>	<b>Dir</b>	<b>Clock Domain</b>	<b>Description</b>
TXPRBSSEL[2:0]	In	TXUSRCLK2	<p>Transmitter PRBS generator test pattern control.</p> <p>000: Standard operation mode (test pattern generation is off)</p> <p>001: PRBS-7</p> <p>010: PRBS-15</p> <p>011: PRBS-23</p> <p>100: PRBS-31</p> <p>101: PCI Express compliance pattern. Only works with 20-bit and 40-bit modes</p> <p>110: Square wave with 2 UI (alternating 0s/1s)</p> <p>111: Square wave with 16 UI or 20 UI period (based on data width)</p>
TXPRBSFORCEERR	In	TXUSRCLK2	When this port is driven High, errors are forced in the PRBS transmitter. While this port is asserted, the output data pattern contains errors. When TXPRBSSEL is set to 000, this port does not affect TXDATA.

Table 3-21 defines the pattern generator attribute.

**Table 3-21: Pattern Generator Attribute**

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
RXPRBS_ERR_LOOPBACK	1-bit Binary	When set to 1, causes RXPRBSERR bit to be internally looped back to TXPRBSFORCEERR of the same GTP transceiver. This allows synchronous and asynchronous jitter tolerance testing without worrying about data clock domain crossing. When set to 0, TXPRBSFORCEERR forces onto the TX PRBS.

## Use Models

The pattern generation and check function are usually used for verifying link quality tests and also for jitter tolerance tests. For link quality testing, the test pattern is chosen by setting TXPRBSSEL and RXPRBSSEL to a non-000 value, and RXPRBS\_ERR\_LOOPBACK is set to 0 (Figure 3-18). Only the PRBS pattern is recognized by the RX pattern checker.

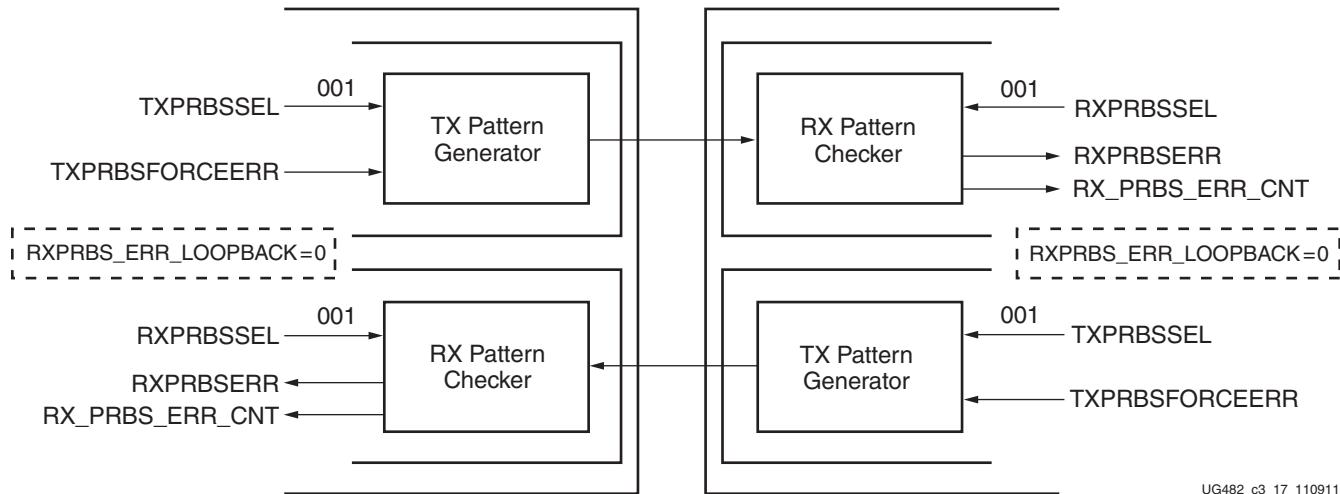


Figure 3-18: Link Test Mode with a PRBS-7 Pattern

To calculate accurately the receiver's bit error rate (BER), an external jitter tolerance tester should be used. For the test, the GTP transceiver should loop the received error status back through the transmitter by setting RXPRBS\_ERR\_LOOPBACK to 1 (Figure 3-19). The same setting should be applied to RXPRBSSEL and TXPRBSSEL.

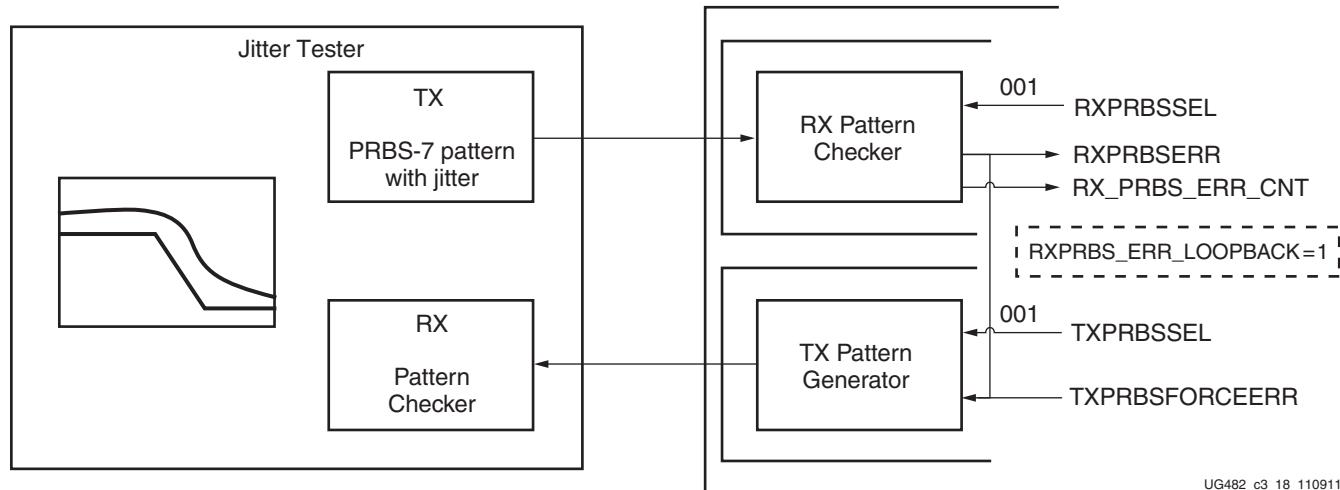


Figure 3-19: Jitter Tolerance Test Mode with a PRBS-7 Pattern

## TX Polarity Control

### Functional Description

If TXP and TXN differential traces are accidentally swapped on the PCB, the differential data transmitted by the GTP transceiver TX is reversed. One solution is to invert the parallel data before serialization and transmission to offset the reversed polarity on the differential pair. The TX polarity control can be accessed through the TXPOLARITY input from the fabric user interface. It is driven High to invert the polarity of outgoing data.

## Ports and Attributes

[Table 3-22](#) defines the ports required for TX polarity control.

**Table 3-22: TX Polarity Control Ports**

Port	Dir	Clock Domain	Description
TXPOLARITY	In	TXUSRCLK2	The TXPOLARITY port is used to invert the polarity of outgoing data. 0: Not inverted. TXP is positive, and TXN is negative. 1: Inverted. TXP is negative, and TXN is positive.

## Using TX Polarity Control

TXPOLARITY can be tied High if the polarity of TXP and TXN needs to be reversed.

# TX Fabric Clock Output Control

## Functional Description

The TX Clock Divider Control block has two main components: serial clock divider control and parallel clock divider and selector control. The clock divider and selector details are illustrated in [Figure 3-20](#).

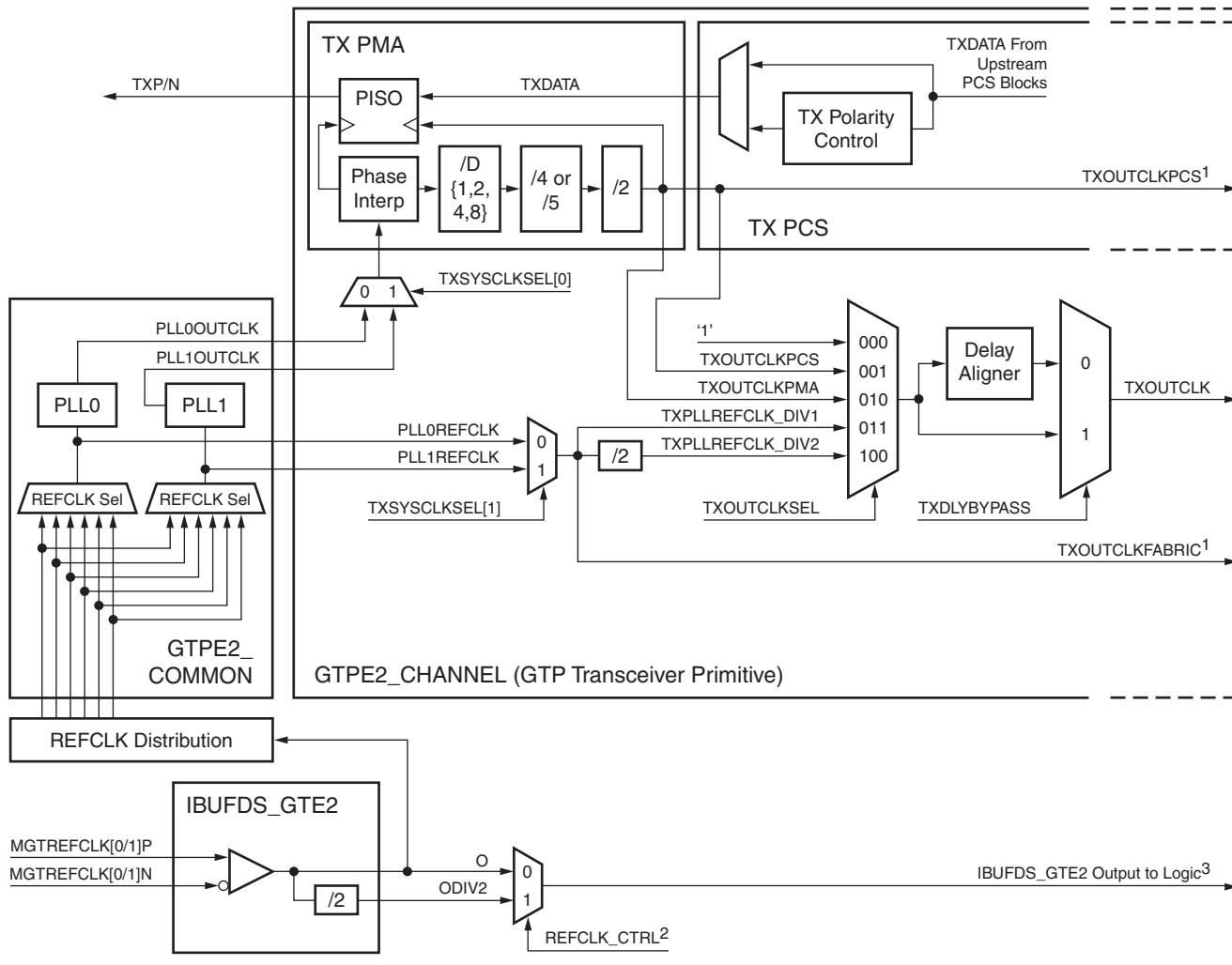


Figure 3-20: TX Serial and Parallel Clock Divider

Notes relevant to Figure 3-20:

1. TXOUTCLKPCS and TXOUTCLKFABRIC are redundant outputs. Use TXOUTCLK for new designs.
2. The REFCLK\_CTRL option is controlled automatically by software and is not user selectable. The user can only route one of the IBUFDS\_GTE2's O or ODIV2 outputs to the FPGA logic.
3. IBUFDS\_GTE2 is a redundant output for additional clocking scheme flexibility.
4. The selection of the /4 or /5 divider block is controlled by the TX\_DATA\_WIDTH attribute from the GTPE2\_CHANNEL primitive. /4 is selected when TX\_DATA\_WIDTH = 16 or 32. /5 is selected when TX\_DATA\_WIDTH = 20 or 40.
5. For details about placement constraints and restrictions on clocking resources (MMCME2, PLLE2, BUFGCTRL, IBUFDS\_GTE2, BUFG, etc.), refer to the [UG472, 7 Series FPGAs Clocking Resources User Guide](#).

## Serial Clock Divider

Each transmitter PMA module has a D divider that divides down the clock from the PLL for lower line rate support. This serial clock divider, D, can be set statically for applications with a fixed line rate or it can be changed dynamically for protocols with multiple line rates.

To use the D divider in fixed line rate applications, the TXOUT\_DIV attribute must be set to the appropriate value, and the TXRATE port needs to be tied to 3'b000. Refer to the Static Setting via Attribute column in [Table 3-23](#) for details.

To use the D divider in multiple line rate applications, the TXRATE port is used to dynamically select the D divider value. The TXOUT\_DIV attribute and the TXRATE port must select the same D divider value upon device configuration. After device configuration, the TXRATE is used to dynamically change the D divider value. Refer to the Dynamic Control via Ports column in [Table 3-23](#) for details.

The control for the serial divider is shown in [Table 3-23](#). For details about the line rate range per speed grade, refer to the [7 series FPGAs documentation page](#) for the appropriate data sheet.

**Table 3-23: TX PLL Output Divider Setting**

D Divider Value	Static Setting via Attribute	Dynamic Control via Ports
1	TXOUT_DIV = 1 TXRATE = 3'b000	TXOUT_DIV = Ignored TXRATE = 3'b001
2	TXOUT_DIV = 2 TXRATE = 3'b000	TXOUT_DIV = Ignored TXRATE = 3'b010
4	TXOUT_DIV = 4 TXRATE = 3'b000	TXOUT_DIV = Ignored TXRATE = 3'b011
8	TXOUT_DIV = 8 TXRATE = 3'b000	TXOUT_DIV = Ignored TXRATE = 3'b100

## Parallel Clock Divider and Selector

The parallel clock outputs from the TX clock divider control block can be used as a fabric logic clock, depending on the line rate requirement.

The recommended clock for the fabric is the TXOUTCLK from one of the GTP transceivers. It is also possible to bring the MGTREFCLK directly to the FPGA logic and use as the fabric clock. TXOUTCLK is preferred for general applications as it has an output delay control used for applications that bypass the TX buffer for output lane deskewing or constant datapath delay. Refer to [TX Pattern Generator, page 105](#) for more details.

The TXOUTCLKSEL port controls the input selector and allows these clocks to be output via the TXOUTCLK port:

- TXOUTCLKSEL = 3'b001: The TXOUTCLKPCS path is not recommended for use because it incurs extra delay from the PCS block.
- TXOUTCLKSEL = 3'b010: TXOUTCLKPMA is the divided down PLL clock after the TX phase interpolator and is used by the TX PCS block. This clock is interrupted when the PLL is reset by one of the related reset signals.
- TXOUTCLKSEL = 3'b011 or 3'b100: TXPLLREFCLK\_DIV1 or TXPLLREFCLK\_DIV2 is the input reference clock to PLL0 or PLL1, depending on the

TXSYSCLKSEL[1] setting. TXPLLREFCLK is the recommended clock for general usage and is required for the TX buffer bypass mode.

## Ports and Attributes

[Table 3-24](#) defines the ports required for TX fabric clock output control.

**Table 3-24: TX Fabric Clock Output Control Ports**

Port	Dir	Clock Domain	Description
TXOUTCLKSEL[2:0]	In	Async	This port controls the multiplexer select signal in <a href="#">Figure 3-20</a> . 3'b000: Static 1 3'b001: TXOUTCLKPCS path 3'b010: TXOUTCLKPMA path 3'b011: TXPLLREFCLK_DIV1 path 3'b100: TXPLLREFCLK_DIV2 path Others: Reserved.
TXRATE[2:0]	In	TXUSRCLK2 (TXRATEMODE makes this port asynchronous)	This port dynamically controls the setting for the TX serial clock divider D (see <a href="#">Table 3-23</a> ), and it is used with the TXOUT_DIV attribute. 3'b000: Use the TXOUT_DIV divider value 3'b001: Set the D divider to 1 3'b010: Set the D divider to 2 3'b011: Set the D divider to 4 3'b100: Set the D divider to 8
TXOUTCLKFABRIC	Out	Clock	TXOUTCLKFABRIC is a redundant output reserved for testing. TXOUTCLK with TXOUTCLKSEL = 3'b011 should be used instead.
TXOUTCLK	Out	Clock	TXOUTCLK is the recommended clock output to the FPGA logic. The TXOUTCLKSEL port is the input selector for TXOUTCLK and allows the PLL input reference clock to the FPGA logic.
TXOUTCLKPCS	Out	Clock	TXOUTCLKPCS is a redundant output. TXOUTCLK with TXOUTCLKSEL = 3'b001 should be used instead.
TXRATEDONE	Out	TXUSRCLK2	The TXRATEDONE port is asserted High for one TXUSRCLK2 cycle in response to a change on the TXRATE port. The TRANS_TIME_RATE attribute defines the period of time between a change on the TXRATE port and the assertion of TXRATEDONE.

**Table 3-24: TX Fabric Clock Output Control Ports (Cont'd)**

<b>Port</b>	<b>Dir</b>	<b>Clock Domain</b>	<b>Description</b>
TXDLYBYPASS	In	Async	TX delay alignment bypass: 0: Uses the TX delay alignment circuit. Set to 1'b0 when the TX buffer is bypassed. 1: Bypasses the TX delay alignment circuit. Set to 1'b1 when the TX buffer is used.
TXRATEMODE	In	Async	Determines if TXRATE should be treated as synchronous or asynchronous. 0: Synchronous. When set to 1'b0, an automatic reset sequence occurs in response to a change on the TXRATE port. 1: Asynchronous.

Table 3-25 defines the attributes required for TX fabric clock output control.

**Table 3-25: TX Fabric Clock Output Control Attributes**

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
TRANS_TIME_RATE	8-bit Hex	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. This attribute determines when PHYSTATUS and TXRATEDONE are asserted after a rate change.
TXBUF_RESET_ON_RATE_CHANGE	String	When set to TRUE, this attribute enables an automatic TX buffer reset during a rate change event initiated by a change in TXRATE.
TXOUT_DIV	Integer	This attribute controls the setting for the TX serial clock divider. This attribute is only valid when TXRATE = 3'b000. Otherwise the D divider value is controlled by TXRATE. Valid settings are 1, 2, 4, and 8.

## TX Phase Interpolator PPM Controller

### Functional Description

The TX Phase Interpolator Parts Per Million (TXPIPPM) Controller module provides support for dynamically controlling the TX phase interpolator (TX PI). Located in the TX PCS, its inputs come from the FPGA TX Interface and it outputs to the TX PMA.

Applications exist that require fine-tune control of the data in the TX PMA. Control of the output clock from the PLL is achieved through a TX PI, which in turn can be controlled by the TX phase interpolator PPM controller module. The FPGA logic can control the TX PI in the TX PMA through the use of the TX phase interpolator PPM controller module in the PCS.

## Ports and Attributes

**Table 3-26** defines the ports required for the TX phase interpolator PPM controller.

**Table 3-26: TX Phase Interpolator PPM Controller Ports**

Port	Dir	Clock Domain	Description
TXPIPPMEN	In	TXUSRCLK2	1 'b0: Disables the TX Phase Interpolator PPM Controller block. The TX PI is not updated with a PI code and retains the previous PI code. 1 'b1: Enables the TX Phase Interpolator PPM Controller block. The TX PI is updated with a PI code every TXPI_SYNREQ_PPM[2:0] cycles.
TXPIPPMOVRDEN	In	TXUSRCLK2	1 'b0: Normal operation. 1 'b1: Enables direct control of the PI code output to the TX PI in the TX PMA. Use with TXPPMOVRD_VALUE[6:0] to program the value of PI code.
TXPIPPMSEL	In	TXUSRCLK2	Reserved. This should always be tied to 1 'b1.
TXPIPPMPD	In	Async	1 'b0: Does not power down the TX phase interpolator PPM controller module. 1 'b1: Powers down the TX phase interpolator PPM controller module.
TXPIPPMSTEPSENSE[4:0]	In	TXUSRCLK2	TXPIPPMSTEPSENSE[4]: 1 'b1: Increments PI code 1 'b0: Decrements PI code TXPIPPMSTEPSENSE[3:0] is the amount to increment or decrement PI code. Its values range from 0 to 15.

[Table 3-27](#) defines the attributes required for the TX phase interpolator PPM controller.

**Table 3-27: TX Phase Interpolator PPM Controller Attributes**

Attribute	Type	Description
TXPI_SYNFFREQ_PPM[2:0]	3-bit Binary	This attribute specifies how often PI code to the TX PI is updated. It is updated every (TXPI_SYNFFREQ_PPM[2:0] + 1) cycles. All values are valid except for 3'b000. The GT Wizard's default value should be used for this attribute.
TXPI_PPM_CFG[7:0]	8-bit Binary	When TXPIPPMOVREN = 1'b1, the lower 7 bits of this attribute should be programmed to one of the 128 values output to the TX PI. The most significant bit needs to be pulsed (asserted High and then Low) for the TX PI to register the new 7-bit value of TXPI_PPM_CFG[6:0].
TXPI_INVSTROBE_SEL	1-bit Binary	Reserved. Tied to 1'b0.
TXPI_GREY_SEL	1-bit Binary	1'b0: TXPIPPMSTEPSENSE[3:0] is binary encoded. 1'b1: TXPIPPMSTEPSENSE[3:0] is grey encoded.
TXPI_PPMCLK_SEL	String	Reserved. The GT Wizard's default value should be used for this attribute.

## TX Phase Interpolator PPM Controller Use Mode

The following describes a sample use case:

- A frequency counter in the fabric determines the lead/lag relationship between the two clocks of interest and increments or decrements (TXPIPPMSTEPSENSE[4]) PI code by a certain step size (TXPIPPMSTEPSENSE[3:0]).
- A sampler and lock detect circuit in the fabric determines when the two clocks are phase aligned. When not phase aligned, the user asserts a signal to the TX phase interpolator PPM controller with the desirable PI code.

This continual phase shifting (fine-tuning) occurs when the lock detect circuit deems the two clocks out of phase and enables the TX phase interpolator PPM controller.

# TX Configurable Driver

## Functional Description

The GTP transceiver TX driver is a high-speed current-mode differential output buffer. To maximize signal integrity, it includes these features:

- Differential voltage control
- Pre-cursor and post-cursor transmit pre-emphasis
- Calibrated termination resistors

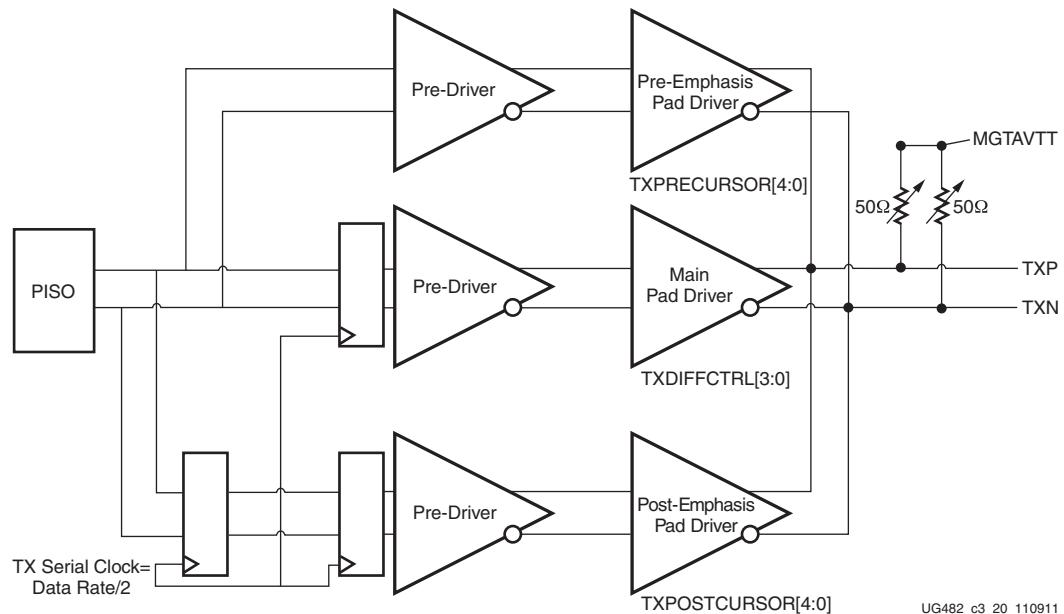


Figure 3-21: TX Configurable Driver Block Diagram

## Ports and Attributes

Table 3-28 defines the TX configurable driver ports.

Table 3-28: TX Configurable Driver Ports

Port	Dir	Clock Domain	Description
TXBUFDIFFCTRL[2:0]	In	Async	Pre-driver Swing Control. The default is 3'b100 (nominal value). Do <i>not</i> modify this value.
TXDEEMPH	In	TXUSRCLK2	TX de-emphasis control for PCI Express PIPE 2.0 interface. This signal is mapped internally to TXPOSTCURSOR via attributes. 0: 6.0 dB de-emphasis (TX_DEEMPH_0[4:0] attribute) 1: 3.5 dB de-emphasis (TX_DEEMPH_1[4:0] attribute)

Table 3-28: TX Configurable Driver Ports (Cont'd)

Port	Dir	Clock Domain	Description																																		
TXDIFFCTRL[3:0]	In	Async	<p>Driver Swing Control. The default is user specified. All listed values are in mV<sub>PPD</sub>.</p> <table border="1"> <thead> <tr> <th>[3:0]</th><th>mV<sub>PPD</sub></th></tr> </thead> <tbody> <tr><td>4'b0000</td><td>253</td></tr> <tr><td>4'b0001</td><td>316</td></tr> <tr><td>4'b0010</td><td>377</td></tr> <tr><td>4'b0011</td><td>439</td></tr> <tr><td>4'b0100</td><td>499</td></tr> <tr><td>4'b0101</td><td>561</td></tr> <tr><td>4'b0110</td><td>621</td></tr> <tr><td>4'b0111</td><td>682</td></tr> <tr><td>4'b1000</td><td>743</td></tr> <tr><td>4'b1001</td><td>799</td></tr> <tr><td>4'b1010</td><td>857</td></tr> <tr><td>4'b1011</td><td>909</td></tr> <tr><td>4'b1100</td><td>959</td></tr> <tr><td>4'b1101</td><td>1002</td></tr> <tr><td>4'b1110</td><td>1043</td></tr> <tr><td>4'b1111</td><td>1074</td></tr> </tbody> </table>	[3:0]	mV <sub>PPD</sub>	4'b0000	253	4'b0001	316	4'b0010	377	4'b0011	439	4'b0100	499	4'b0101	561	4'b0110	621	4'b0111	682	4'b1000	743	4'b1001	799	4'b1010	857	4'b1011	909	4'b1100	959	4'b1101	1002	4'b1110	1043	4'b1111	1074
[3:0]	mV <sub>PPD</sub>																																				
4'b0000	253																																				
4'b0001	316																																				
4'b0010	377																																				
4'b0011	439																																				
4'b0100	499																																				
4'b0101	561																																				
4'b0110	621																																				
4'b0111	682																																				
4'b1000	743																																				
4'b1001	799																																				
4'b1010	857																																				
4'b1011	909																																				
4'b1100	959																																				
4'b1101	1002																																				
4'b1110	1043																																				
4'b1111	1074																																				
TXELECIDLE	In	TXUSRCLK2	When High, this signal forces GTPTXP and GTPTXN both to Common mode, creating an electrical idle signal.																																		
TXINHIBIT	In	TXUSRCLK2	When High, this signal blocks transmission of TXDATA and forces GTPTXP to 0 and GTPTXN to 1.																																		
TXMAINCURSOR[6:0]	In	Async	<p>Allows the main cursor coefficients to be directly set if the TX_MAINCURSOR_SEL attribute is set to 1'b1.</p> <p>51 – TXPOSTCURSOR coefficient units – TXPRECURSOR coefficient units  <math>\leq</math> TXMAINCURSOR coefficient units  <math>\leq</math> 80 – TXPOSTCURSOR coefficient units – TXPRECURSOR coefficient units.</p>																																		

Table 3-28: TX Configurable Driver Ports (Cont'd)

Port	Dir	Clock Domain	Description								
			[2:0]	Full Range	Half Range	Full Range Attribute	Half Range Attribute				
TXMARGIN[2:0]	In	Async	TX Margin control for PCI Express PIPE 2.0 Interface. These signals are mapped internally to TXDIFFCTRL/TXBUDIFFCTRL via attributes.	000	800-1200	400-1200	TX_MARGIN_FULL_0	TX_MARGIN_LOW_0			
			001	800-1200	400-700	TX_MARGIN_FULL_1	TX_MARGIN_LOW_1				
			010	800-1200	400-700	TX_MARGIN_FULL_2	TX_MARGIN_LOW_2				
			011	200-400	100-200	TX_MARGIN_FULL_3	TX_MARGIN_LOW_3				
			100	100-200	100-200	TX_MARGIN_FULL_4	TX_MARGIN_LOW_4				
			101	default to "DIRECT" mode							
			110								
			111								
PMARSVDIN1	In	Async	Reserved.								
PMARSVDIN0	In	Async	Reserved.								

Table 3-28: TX Configurable Driver Ports (Cont'd)

Port	Dir	Clock Domain	Description		
TXPOSTCURSOR[4:0]	In	Async	Transmitter post-cursor TX pre-emphasis control. The default is user specified. All listed values (dB) are typical.		
			[4:0]	Emphasis (dB)	Coefficient Units
			5'b00000	0.00	0
			5'b00001	0.22	1
			5'b00010	0.45	2
			5'b00011	0.68	3
			5'b00100	0.92	4
			5'b00101	1.16	5
			5'b00110	1.41	6
			5'b00111	1.67	7
			5'b01000	1.94	8
			5'b01001	2.21	9
			5'b01010	2.50	10
			5'b01011	2.79	11
			5'b01100	3.10	12
			5'b01101	3.41	13
			5'b01110	3.74	14
			5'b01111	4.08	15
			5'b10000	4.44	16
			5'b10001	4.81	17
			5'b10010	5.19	18
			5'b10011	5.60	19
			5'b10100	6.02	20
			5'b10101	6.47	21
			5'b10110	6.94	22
			5'b10111	7.43	23
			5'b11000	7.96	24
			5'b11001	8.52	25
			5'b11010	9.12	26
			5'b11011	9.76	27
			5'b11100	10.46	28
			5'b11101	11.21	29
			5'b11110	12.04	30
			5'b11111	12.96	31
TXPOSTCURSORINV	In	Async	When set to 1'b1, inverts the polarity of the TXPOSTCURSOR coefficient. The default is 1'b0.		

Table 3-28: TX Configurable Driver Ports (Cont'd)

Port	Dir	Clock Domain	Description		
TXPRECURSOR[4:0]	In	Async	Transmitter pre-cursor TX pre-emphasis control. The default is user specified. All listed values (dB) are typical.		
			[4:0]	Emphasis (dB)	Coefficient Units
			5'b00000	0.00	0
			5'b00001	0.22	1
			5'b00010	0.45	2
			5'b00011	0.68	3
			5'b00100	0.92	4
			5'b00101	1.16	5
			5'b00110	1.41	6
			5'b00111	1.67	7
			5'b01000	1.94	8
			5'b01001	2.21	9
			5'b01010	2.50	10
			5'b01011	2.79	11
			5'b01100	3.10	12
			5'b01101	3.41	13
			5'b01110	3.74	14
			5'b01111	4.08	15
			5'b10000	4.44	16
			5'b10001	4.81	17
			5'b10010	5.19	18
			5'b10011	5.60	19
			5'b10100	6.02	20
			5'b10101	6.02	20
			5'b10110	6.02	20
			5'b10111	6.02	20
			5'b11000	6.02	20
			5'b11001	6.02	20
			5'b11010	6.02	20
			5'b11011	6.02	20
			5'b11100	6.02	20
			5'b11101	6.02	20
			5'b11110	6.02	20
			5'b11111	6.02	20
TXPRECURSORINV	In	Async	When set to 1'b1, inverts the polarity of the TXPRECURSOR coefficient. The default is 1'b0.		

Table 3-28: TX Configurable Driver Ports (Cont'd)

Port	Dir	Clock Domain	Description
MGTPTXP MGTPTXN	Out (Pad)	TX Serial Clock	MGTPTXP and MGTPTXN are differential complements of one another forming a differential transmit output pair. These ports represent the pads. The locations of these ports must be constrained (see <a href="#">Implementation, page 20</a> ) and brought to the top level of the design.
TXSWING	In	Async	TX swing control for PCI Express PIPE 2.0 Interface. This signal is mapped internally to TXDIFFCTRL/TXBUFDIFFCTRL. 0: Full swing 1: Low swing

Table 3-29 defines the TX configurable driver attributes.

Table 3-29: TX Configurable Driver Attributes

Attribute	Type	Description
TX_DEEMPH0[4:0]	5-bit Binary	This attribute has the value of TXPOSTCURSOR[4:0] that has to be mapped when TXDEEMPH = 0. TX_DEEMPH_0[4:0] = TXPOSTCURSOR[4:0]. The default is 5'b10100. Do not modify this value.
TX_DEEMPH1[4:0]	5-bit Binary	This attribute has the value of TXPOSTCURSOR[4:0] that has to be mapped when TXDEEMPH = 1. TX_DEEMPH_1[4:0] = TXPOSTCURSOR[4:0]. The default is 5'b01101. Do not modify this value.
TX_DRIVE_MODE	String	This attribute selects whether PCI Express PIPE 2.0 ports or TX Drive Control ports control the TX driver. The default is "DIRECT". DIRECT: TXBUFDIFFCRL, TXDIFFCTRL, TXPOSTCURSOR, TXPRECURSOR and TXMAINCURSOR (If TX_MAINCURSOR_SEL = 1'b1) control the TX driver settings. PIPE: TXDEEMPH, TXMARGIN, TXSWING, TXPRECURSOR and TXMAINCURSOR (If TX_MAINCURSOR_SEL = 1'b1) control the TX driver settings.
TX_MAINCURSOR_SEL	1-bit Binary	Allows independent control of the main cursor. 1'b0: The TXMAINCURSOR coefficient is automatically determined by the equation: 80 – TXPOSTCURSOR coefficient – TXPRECURSOR coefficient 1'b1: TXMAINCURSOR coefficient can be independently set by the TXMAINCURSOR port within the range specified in the pin description.
TX_MARGIN_FULL_0[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 000 and TXSWING = 0. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1001111 (1000 mV <sub>PPD</sub> typical).

Table 3-29: TX Configurable Driver Attributes (Cont'd)

Attribute	Type	Description
TX_MARGIN_FULL_1[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 001 and TXSWING = 0. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1001111 (1000 mV <sub>PPD</sub> typical).
TX_MARGIN_FULL_2[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 010 and TXSWING = 0. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1001111 (1000 mV <sub>PPD</sub> typical).
TX_MARGIN_FULL_3[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 0011 and TXSWING = 0. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1000001 (300 mV <sub>PPD</sub> typical).
TX_MARGIN_FULL_4[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 100 and TXSWING = 0. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1000000 (250 mV <sub>PPD</sub> typical).
TX_MARGIN_LOW_0[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 000 and TXSWING = 1. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1000111 (600 mV <sub>PPD</sub> typical).
TX_MARGIN_LOW_1[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 001 and TXSWING = 1. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1000110 (550 mV <sub>PPD</sub> typical).
TX_MARGIN_LOW_2[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 010 and TXSWING = 1. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1000100 (450 mV <sub>PPD</sub> typical).
TX_MARGIN_LOW_3[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 0011 and TXSWING = 1. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1000000 (250 mV <sub>PPD</sub> typical).

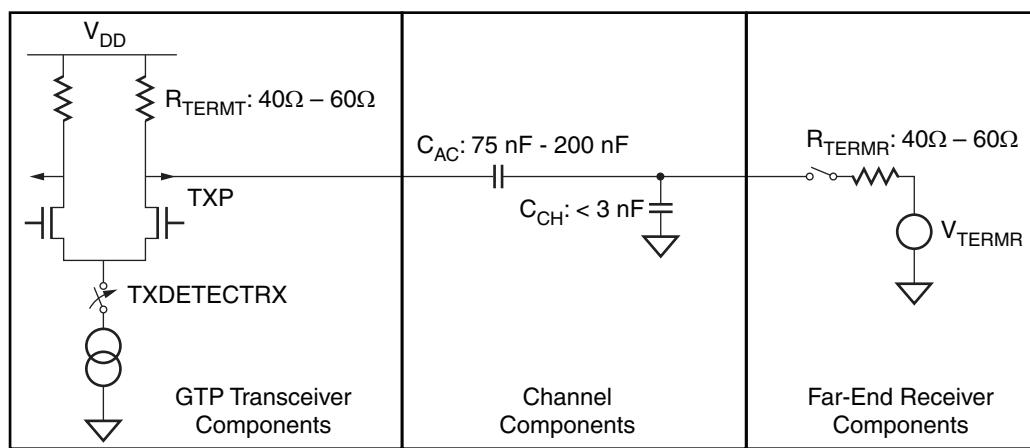
Table 3-29: TX Configurable Driver Attributes (Cont'd)

Attribute	Type	Description
TX_MARGIN_LOW_4[6:0]	7-bit Binary	This attribute has the value of TXBUFDIFFCTRL[2:0] and TXDIFFCTRL[3:0] that has to be mapped when TXMARGIN = 100 and TXSWING = 1. TX_MARGIN_FULL_0 = TXBUFDIFFCTRL[2:0], TXDIFFCTRL[3:0]. The default is 7'b1000000 (250 mV <sub>PPD</sub> typical).
TX_PREDRIVER_MODE	1-bit Binary	This is a restricted attribute. Always set this to 1'b0. Do not modify this attribute.
PMA_RSV5	1-bit Binary	Reserved.

## TX Receiver Detect Support for PCI Express Designs

### Functional Description

The PCI Express specification includes a feature that allows the transmitter on a given link to detect if a receiver is present. The decision if a receiver is present is based on the rise time of TXP/TXN. Figure 3-22 shows the circuit model used for receive detection. The GTP transceiver must be in the P1 power down state to perform receiver detection. Receiver detection requires a 75 nF to 200 nF external coupling capacitor between the transmitter and receiver, and the receiver must be terminated to GND. The receiver detection sequence starts with the assertion of TXDETECTRX. In response, the receiver detection logic drives TXN and TXP to ( $V_{DD} - V_{SWING}/2$ ) and then releases them. After a programmable interval, the levels of TXN and TXP are compared with a threshold voltage. At the end of the sequence, the receiver detection status is presented on RXSTATUS when PHYSTATUS is asserted High for one cycle.



UG482\_c3\_21\_110911

Figure 3-22: Receiver Detection Circuit Model

### Ports and Attributes

Table 3-30 describes the TX receiver detection ports.

Table 3-30: TX Receiver Detection Ports

Port	Dir	Clock Domain	Description
TXDETECTRX	In	TXUSRCLK2	Used to tell the GTP transceiver to begin a receiver detection operation. 0: Normal operation. 1: Receiver detection.
TXPD[1:0]	In	TXUSRCLK2	
RXPD[1:0]	In	RXUSRCLK2	Power up or down the TX and RX of the GTP transceiver. In PCI Express mode, TXPD and RXPD should be tied to the same source. To perform receiver detection, set these signals to the P1 power saving state. 00: P0 power state for normal operation. 01: P0s power saving state with low recovery time latency. 10: P1 power saving state with longer recovery time latency. 11: P2 power saving state with lowest power.
PHYSTATUS	Out	RXUSRCLK2	In PCI Express mode, this signal is used to communicate completion of several GTP transceiver functions, including power management state transitions, rate change, and receiver detection. During receiver detection, this signal is asserted High to indicate receiver detection completion.
RXSTATUS[2:0]	Out	RXUSRCLK2	During receiver detection, this signal is read when PHYSTATUS is asserted High. Only these encodings are valid during receiver detection: 000: Receiver not present. 011: Receiver present.

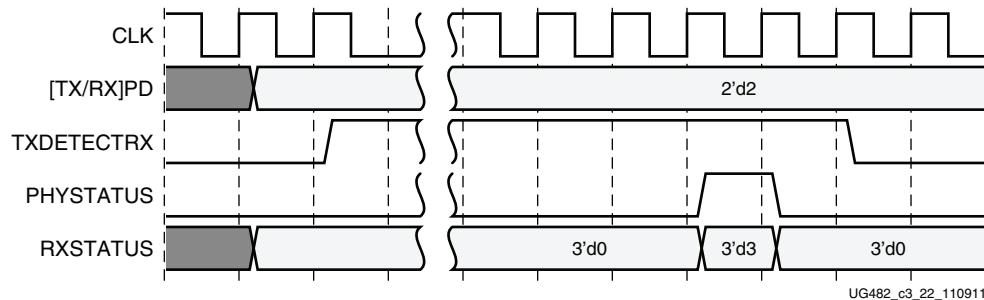
Table 3-31: TX Receiver Detection Attributes

Attribute	Type	Description
TX_RXDETECT_CFG	14-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
TX_RXDETECT_REF	3-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

## Using the TX Receiver Detection for PCI Express

While in the P1 power state, the GTP transceiver can be instructed to perform a receiver detection operation to determine if there is a receiver at the other end of the link.

Figure 3-23 shows an example use mode on how to perform receiver detection in PCI Express mode.



**Figure 3-23: PCI Express Receiver Detection**

**Note:** Figure 3-23 shows the sequence of events for the receiver present case and is not drawn to scale.

Notes relevant to Figure 3-23:

1. Ensure that the GTP transceiver has successfully entered the P1 power state with  $[TX/RX]PD = 2'd2$  before receiver detection is performed by asserting TXDETECTRX.
2. Wait for  $PHYSTATUS = 1'd1$  to read RXSTATUS on the same PCLK cycle. In PCI Express mode, PCLK is  $[TX/RX]USRCLK$ . If  $RXSTATUS = 3'd3$ , then the receiver is present. If  $RXSTATUS = 3'd0$ , then the receiver is not present. Deassert TXDETECTRX to exit receiver detection.

## TX Out-of-Band Signaling

### Functional Description

Each GTP transceiver provides support for generating the out-of-band (OOB) sequences described in the Serial ATA (SATA), Serial Attach SCSI (SAS) specification, and beaconing described in the PCI Express specification.

### Ports and Attributes

Table 3-32 shows the OOB signaling related ports.

**Table 3-32: TX OOB Signaling Ports**

Port	Dir	Clock Domain	Description
TXCOMFINISH	Out	TXUSRCLK2	Indicates completion of transmission of the last SAS or SATA COM beacon.
TXCOMINIT	In	TXUSRCLK2	Initiates transmission of the COMINIT sequence for SATA/SAS.
TXCOMSAS	In	TXUSRCLK2	Initiates transmission of the COMSAS sequence for SAS.
TXCOMWAKE	In	TXUSRCLK2	Initiates transmission of the COMWAKE sequence for SATA/SAS.

Table 3-32: TX OOB Signaling Ports (Cont'd)

Port	Dir	Clock Domain	Description
TXPDELECIDLEMODE	In	TXUSRCLK2	Determines if TXELECIDLE and TXPD should be treated as synchronous or asynchronous signals. Enables compliance during cold and warm PCI Express resets. 1: Asynchronous 0: Synchronous
TXPD[1:0]	In	TXUSRCLK2	Powers down the TX lane according to the PCI Express encoding. 00: P0 normal operation 01: P0s low recovery time power down 10: P1 longer recovery time, RecDet still on 11: P2 lowest power state. Attributes can control the transition times between these power down mode (PD_TRANS_TIME_FROM_P2, PD_TRANS_TIME_NONE_P2, PD_TRANS_TIME_TO_P2).

Table 3-33 shows the OOB signaling attributes.

Table 3-33: TX OOB Signaling Attributes

Attribute	Type	Description
SATA_PLL_CFG	2-bit Binary	Configuration bits for the PLL setting related to SAS/SATA.
SATA_BURST_SEQ_LEN	4-bit Binary	Number of bursts in a COM sequence for SAS/SATA.
TXOOB_CFG	1-bit Binary	TX OOB configuration.
PCS_RSVD_ATTR[8]	1-bit Binary	OOB Powerdown 1'b0 - circuit powered down 1'b1 - Circuit powered up (PCIe, SAS/SAS, protocols/applications using OOB)

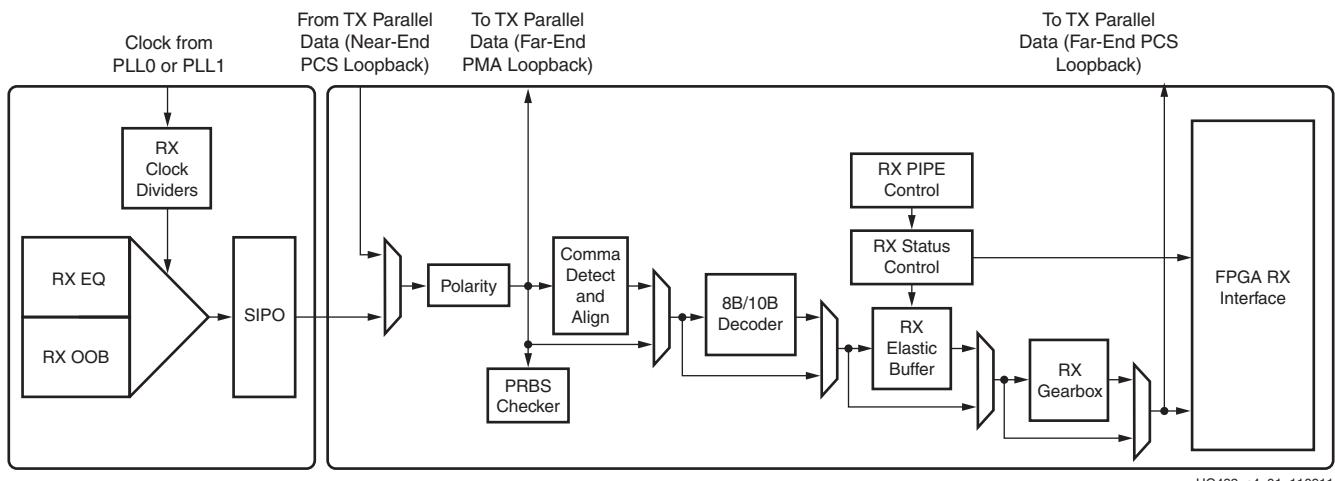
# Receiver

---

## RX Overview

### Functional Description

This section shows how to configure and use each of the functional blocks inside the receiver (RX). Each GTP transceiver includes an independent receiver, made up of a PCS and a PMA. Figure 4-1 shows the blocks of the GTP transceiver RX. High-speed serial data flows from traces on the board into the PMA of the GTP transceiver RX, into the PCS, and finally into the FPGA logic. Refer to Figure 2-9, page 36 for the description of the channel clocking architecture, which provides clocks to the RX and TX clock dividers.



**Figure 4-1: GTP Transceiver RX Block Diagram**

The key elements within the GTP transceiver RX are:

1. RX Analog Front End, page 128
2. RX Out-of-Band Signaling, page 133
3. RX Equalizer, page 141
4. RX CDR, page 143
5. RX Fabric Clock Output Control, page 149
6. RX Margin Analysis, page 153
7. RX Polarity Control, page 161
8. RX Pattern Checker, page 162

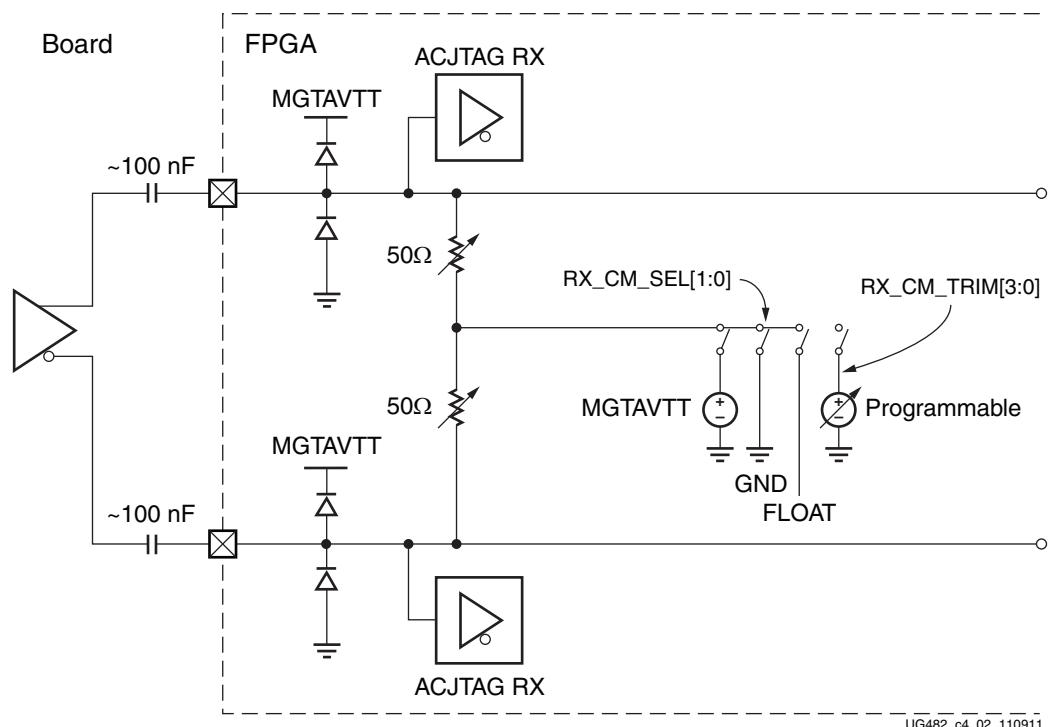
9. RX Byte and Word Alignment, page 164
10. RX 8B/10B Decoder, page 173
11. RX Buffer Bypass, page 177
12. RX Elastic Buffer, page 190
13. RX Clock Correction, page 194
14. RX Channel Bonding, page 201
15. RX Gearbox, page 211
16. FPGA RX Interface, page 218

## RX Analog Front End

### Functional Description

The RX analog front end (AFE) is a high-speed current-mode input differential buffer (see [Figure 4-1](#)). It has these features:

- Configurable RX termination voltage
- Calibrated termination resistors



*Figure 4-2: RX Analog Front End*

## Ports and Attributes

[Table 4-1](#) defines the RX AFE ports.

**Table 4-1: RX AFE Ports**

Port	Dir	Clock Domain	Description
GTPRXN, GTPRXP	In (Pad)	RX Serial Clock	GTPRXN and GTPRXP are differential complements of one another forming a differential receiver input pair. These ports represent pads. The location of these ports must be constrained (see <a href="#">Implementation, page 20</a> ) and brought to the top level of the design.
PMARSVDOUT1	Out	Async	Reserved.
PMARSVDOUT0	Out	Async	Reserved.
PMARSVDIN2	In	Async	Reserved.

[Table 4-2](#) defines the RX AFE attributes.

**Table 4-2: RX AFE Attributes**

Attribute	Type	Description
RX_CM_SEL [1:0]	2-bit Binary	Controls the mode for the RX termination voltage.  2'b00 - AVTT 2'b01 - GND 2'b10 - Floating 2'b11 - Programmable
RX_CM_TRIM [3:0]	4-bit Binary	Controls the Common mode in Programmable mode.  4'b0000 - 100 mV 4'b0001 - 200 mV 4'b0010 - 250 mV 4'b0011 - 300 mV 4'b0100 - 350 mV 4'b0101 - 400 mV 4'b0110 - 500 mV 4'b0111 - 550 mV 4'b1000 - 600 mV 4'b1001 - 700 mV 4'b1010 - 800 mV 4'b1011 - 850 mV 4'b1100 - 900 mV 4'b1101 - 950 mV 4'b1110 - 1000 mV 4'b1111 - 1100 mV

Table 4-2: RX AFE Attributes (Cont'd)

Attribute	Type	Description
TERM_RCAL_CFG[14:0]	15-bit Binary	Controls the internal termination calibration circuit. This feature is intended for internal use only.
TERM_RCAL_OVRD	3-bit Binary	Selects whether the external $100\Omega$ precision resistor connected to the MGTRREF pin or an override value is used, as defined by TERM_RCAL_CFG [14:0]. This feature is intended for internal use only.
RXLPM_INCM_CFG	1-bit Binary	$1'b1$ = Enable high common mode operation $1'b0$ = Disable high common mode operation
RXLPM_IPCM_CFG	1-bit Binary	$1'b1$ = Enable low common mode operation $1'b0$ = Disable low common mode operation

## Use Modes—RX Termination

Table 4-3: Use Mode 1—RX Termination

Use Mode	External AC Coupling	Term Voltage	Max Swing mV <sub>DPP</sub>	Suggested Protocols and Usage Notes
1	On	Gnd	1,200	Attribute Settings: <ul style="list-style-type: none"> <li>• RX_CM_SEL[1:0] = <math>2'b01</math></li> <li>• RXLPM_INCM_CFG = <math>1'b0</math></li> <li>• RXLPM_IPCM_CFG = <math>1'b1</math></li> </ul>

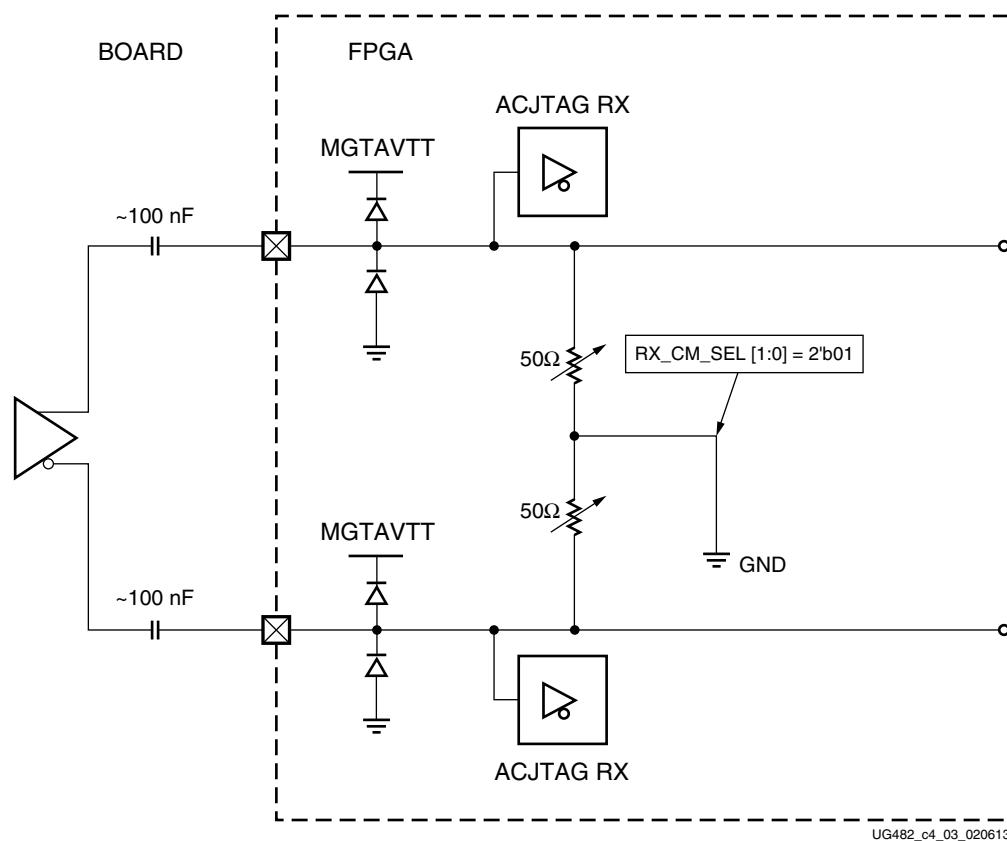


Figure 4-3: Use Mode 1

Table 4-4: Use Mode 2—RX Termination

Use Mode	External AC Coupling	Term Voltage	Max Swing mV <sub>DPP</sub>	Suggested Protocols and Usage Notes
2	On	AVTT	1,200	<p>Protocols:</p> <ul style="list-style-type: none"> <li>• Backplane</li> <li>• CEI-6 (1,200 mV<sub>DPP</sub>)</li> <li>• Wireless</li> <li>• Serial RapidIO</li> </ul> <p>Attribute Settings:</p> <ul style="list-style-type: none"> <li>• RX_CM_SEL[1:0] = 2'b00</li> <li>• RXLPM_INCM_CFG = 1'b1</li> <li>• RXLPM_IPCM_CFG = 1'b0</li> </ul>

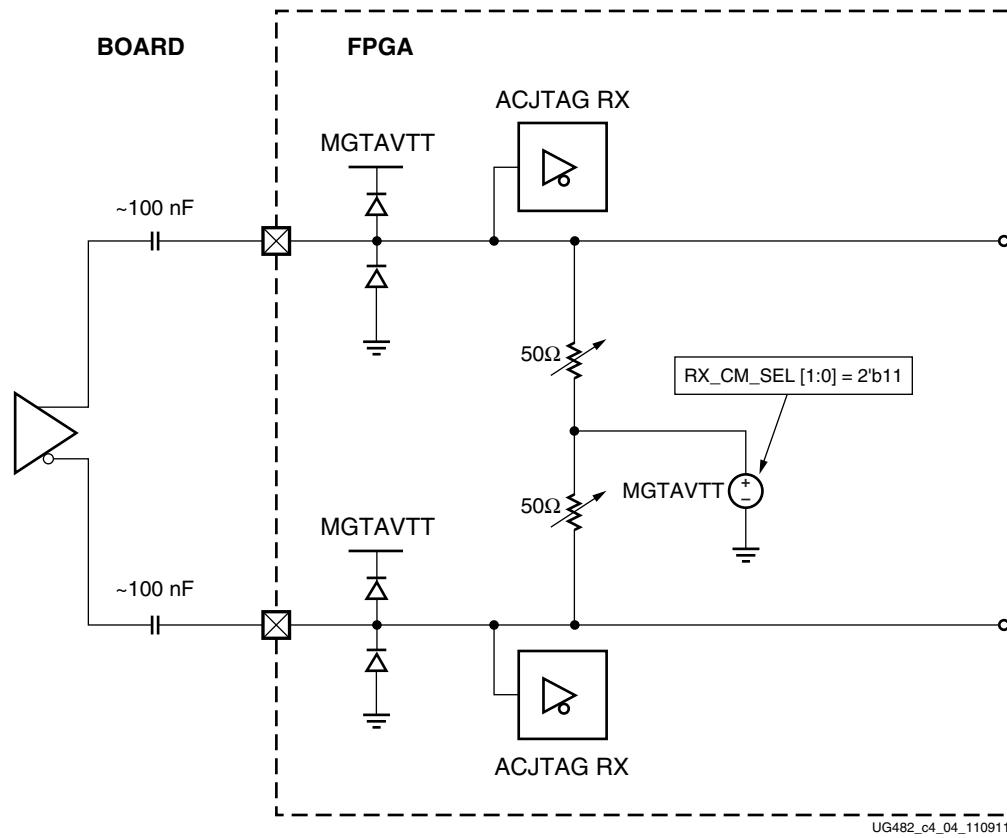


Figure 4-4: Use Mode 2

Table 4-5: Use Mode 3—RX Termination

Use Mode	External AC Coupling	Term Voltage	Max Swing mV <sub>DPP</sub>	Suggested Protocols and Usage Notes
3	On	800 mV	2,000	<p>Protocols:</p> <ul style="list-style-type: none"> <li>• Optical IF (SONET/SDH/OTU)</li> <li>• HD/SD-SDI</li> <li>• XAUI (1,600 mV<sub>DPP</sub>)</li> <li>• GbE</li> <li>• PCIe</li> <li>• Interlaken</li> </ul> <p>Attribute Settings:</p> <ul style="list-style-type: none"> <li>• RX_CM_SEL[1:0] = 2'b11</li> <li>• RXLPM_INCM_CFG = 1'b1</li> <li>• RXLPM_IPCM_CFG = 1'b0</li> <li>• RX_CM_TRIM[3:0] = 4'b1010</li> </ul>

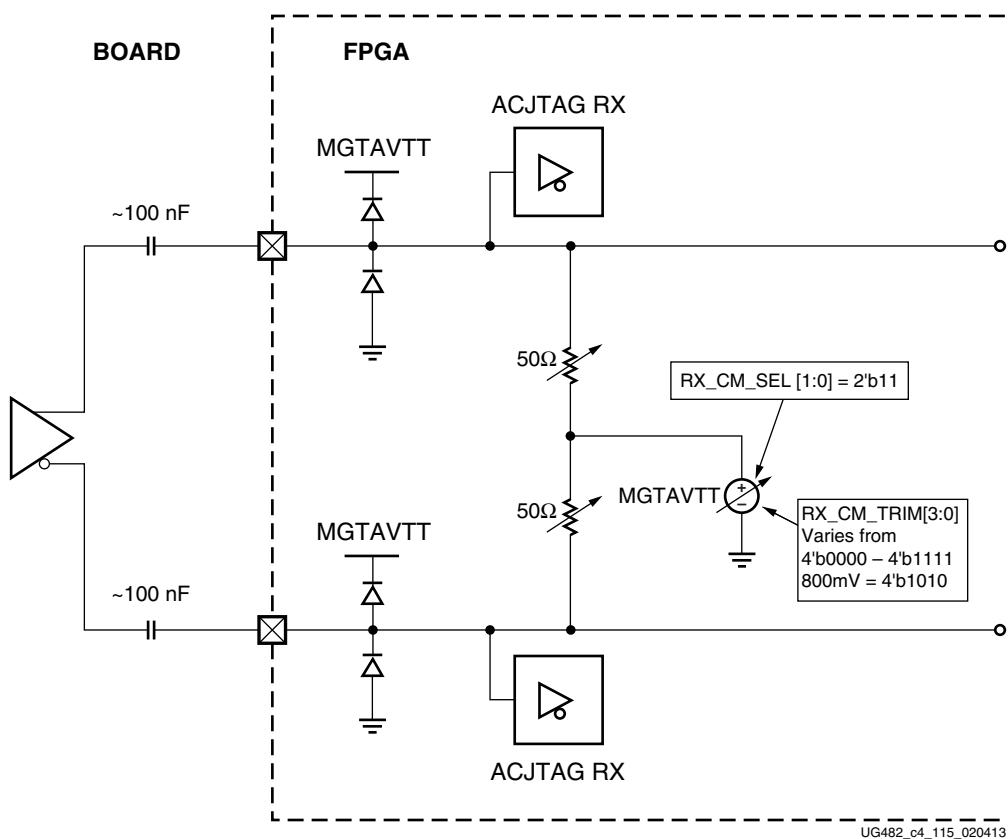


Figure 4-5: Use Mode 3

## RX Out-of-Band Signaling

### Functional Description

The GTP transceiver receiver provides support for decoding the out-of-band (OOB) sequences described in the Serial ATA (SATA) and Serial Attach SCSI (SAS) specifications and supports beaconing described in the PCI Express specification. GTP transceiver receiver support for SATA/SAS OOB signaling consists of the analog circuitry required to decode the OOB signal state and state machines to decode bursts of OOB signals for SATA/SAS COM sequences.

The GTP transceiver receiver also supports beacons that are PCI Express compliant by using interface signals defined in the *PHY Interface for the PCI Express (PIPE) Specification*. The FPGA logic decodes the beacon sequence.

### Ports and Attributes

Table 4-6 defines the OOB signaling related ports.

Table 4-6: RX OOB Signaling Ports

Port	Dir	Clock Domain	Description
RXOOBRESET	In	Async	Reserved. Tie to GND.
RXCOMINITDET	Out	RXUSRCLK2	Indicates reception of the COMINIT sequence for SATA/SAS.
RXCOMSASDET	Out	RXUSRCLK2	Indicates reception of the COMSAS sequence for SAS.
RXCOMWAKEDET	Out	RXUSRCLK2	Indicates reception of the COMWAKE sequence for SATA/SAS.
RXELECIDLE	Out	RXUSRCLK2	This output indicates the status of OOB signal detection and is only valid for PCIe, SATA/SAS, and protocols/applications using OOB. In these cases, the OOB circuit must be powered on. 0 : Activity is seen on the receiver 1 : No activity is seen For non-OOB protocols, RXELECIDLEMODE[1:0] must be set to 2'b11. RXELECIDLE will output a static 1'b0 and in this case does not indicate signal detection status.
RXELECIDLEMODE[1:0]	In	Async	Input signal to control the behavior of RXELECIDLE. 2'b00: RXELECIDLE indicates the status of the OOB signal detection circuit. Use this setting for PCIe, SATA/SAS, and protocols/applications using OOB. In these cases, the OOB circuit must be powered on. 2'b11: RXELECIDLE outputs a static 1'b0. Use this setting for non-OOB protocols.

Table 4-7 defines the OOB signaling attributes.

Table 4-7: RX OOB Signaling Attributes

Attribute	Type	Description
RXOOB_CFG	7-bit Binary	OOB block configuration.
RXOOB_CLK_CFG	String	Selects which clock to use for OOB.
SATA_BURST_VAL	3-bit Binary	Number of bursts to declare a COM match for SAS/SATA.
SATA_EIDLE_VAL	3-bit Binary	Number of idles to declare a COM match for SAS/SATA.
SAS_MIN_COM	Integer	1-63. Lower bound on activity burst for COM FSM for SAS/SATA.

Table 4-7: RX OOB Signaling Attributes (Cont'd)

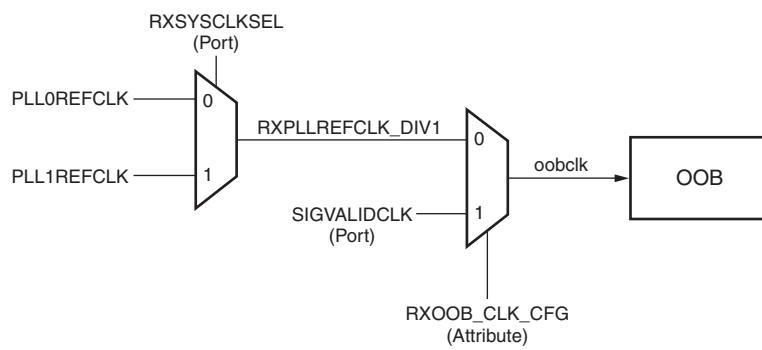
Attribute	Type	Description
SATA_MIN_INIT	Integer	1-63. Lower bound on idle count during COMSAS for SAS.
SATA_MIN_WAKE	Integer	1-63. Lower bound on idle count during COMINIT/COMRESET for SAS/SATA.
SATA_MAX_BURST	Integer	1-63. Upper bound on activity burst for COM FSM for SAS/SATA.
SAS_MAX_COM	Integer	1-127. Upper bound on idle count during COMSAS for SAS.
SATA_MAX_INIT	Integer	1-63. Upper bound on idle count during COMINIT/COMRESET for SAS/SATA.
SATA_MAX_WAKE	Integer	1-63. Upper bound on idle count during COMWAKE for SAS/SATA.
PCS_RSVD_ATTR[8]	1-bit Binary	OOB power up. The OOB circuit can be optionally powered down when not being used. 1'b0 = Circuit powered down. 1'b1 = Circuit powered up (PCIe, SATA/SAS, protocols/applications using OOB).

## Use Mode

To use OOB, the following RX termination conditions need to be applied:

- AC-coupled case: Termination voltage should be 800 mV or greater
- DC-coupled case: Termination voltage should be 900 mV or greater

Also, the attribute PCS\_RSVD\_ATTR[8] should be set to 1'b1. The OOB circuit has two possible sources from which it can receive a clock, as shown in [Figure 4-6](#).



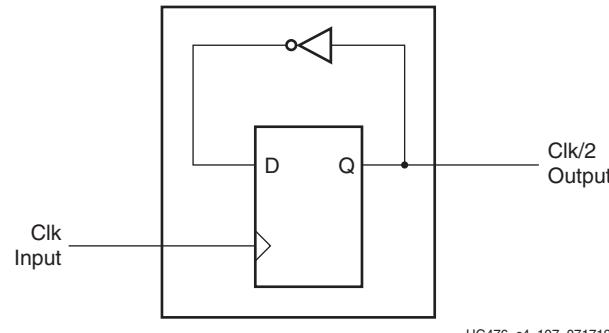
UG482\_c4\_106\_021113

Figure 4-6: Clocking Mechanism for the OOB Detect Circuit

The attribute RXOOB\_CLK\_CFG controls the source of oobclk. Setting RXOOB\_CLK\_CFG to 1'b0 selects the reference clock connected to PLL0 or PLL1. RXSYSCLKSEL controls which of the two reference clocks is selected. Setting RXOOB\_CLK\_CFG to 1'b0 selects an alternative clock source from SIGVALIDCLK. A divided down reference clock can be

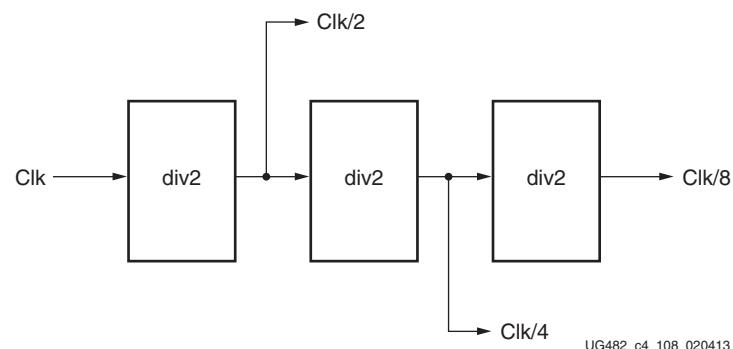
connected to the SIGVALIDCLK port pin, providing an alternative clock for the OOB circuit.

The divided down clock(s) requires no special phase relationships between other clocks in the SERDES. However, there is a requirement of a 50% duty cycle. [Figure 4-7](#) and [Figure 4-8](#) show the method for clock division. [Figure 4-7](#) shows how a simple toggle flip-flop can be used to divide the REFCLK.



**Figure 4-7: Toggle Flip-Flop to Divide REFCLK**

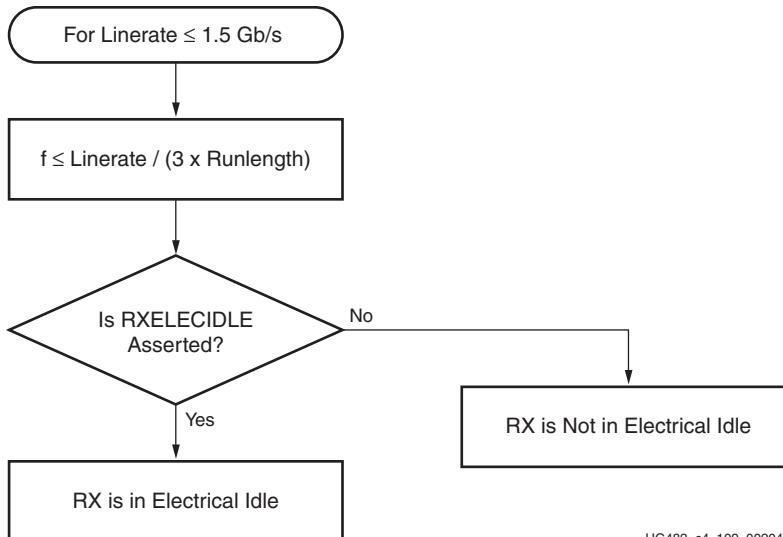
[Figure 4-8](#) shows how cascading several divide-by-two circuits produces higher order clock dividers such as divide-by-4 and divide-by-8.



**Figure 4-8: Clock Dividers**

## Use Modes

For OOB operating at a line rate of 1.5 Gb/s or below, see the flowchart in [Figure 4-9](#) to determine the frequency  $f$  of the OOB clock.



UG482\_c4\_109\_0020413

*Figure 4-9: Flowchart for Protocols with Line Rates  $\leq 1.5\text{G}$*

The requirement in [Equation 4-1](#) must be satisfied for the OOB to work correctly.

$$f \leq \text{linerate}/(3 \times \text{runlength}) \quad \text{Equation 4-1}$$

OOB operating at line rates  $> 1.5\text{ Gb/s}$  is an advanced feature. Operation for certain protocols at higher line rates such as PCIe (Gen1 and Gen2) and SATA are addressed in [Table 4-8](#).

*Table 4-8: OOB Guidelines for Operating Rates above 1.5 Gb/s*

Protocol	Operation
PCIe Gen1	<p>See <a href="#">Figure 4-10</a> for the algorithm to determine whether the RX is in electrical idle.</p> <p>If a scrambler has not been used, the RX electrical idle should not be used for internal detect logic of the hold/reset logic of the LPM, or CDR<sup>(1)</sup>. The user needs to verify received data to decide whether or not an electrical idle state is present i.e., qualification using incoming data is essential in this mode of operation.</p> <p>If a scrambler is used, electrical idle may solely be used to determine whether the RX is in electrical idle.</p>

Table 4-8: OOB Guidelines for Operating Rates above 1.5 Gb/s (Cont'd)

Protocol	Operation
PCIe Gen2	See <a href="#">Figure 4-11</a> for the algorithm to determine whether the RX is in electrical idle. Other methods that can be used for this are shown in <a href="#">Figure 4-12</a> and <a href="#">Figure 4-13</a> . RX electrical idle should not be used for internal detect logic of the hold/reset logic of the LPM, or CDR <sup>(1)</sup> . The user needs to verify received data to decide whether or not an electrical idle state is present i.e., qualification using incoming data is essential in this mode of operation.
SATA 1.5 Gb/s	Use <a href="#">Equation 4-1</a> to derive the appropriate OOB clock (see <a href="#">Figure 4-9</a> ).
SATA 3 Gb/s	See <a href="#">Figure 4-14</a> for the algorithm to determine whether the RX is in electrical idle. RX electrical idle should not be used for internal detect logic of the hold/reset logic of the LPM, or CDR <sup>(1)</sup> . The user needs to verify received data to decide whether or not an electrical idle state is present i.e., qualification using incoming data is essential in this mode of operation.
SATA 6 Gb/s	See <a href="#">Figure 4-14</a> for the algorithm to determine whether the RX is in electrical idle. RX electrical idle should not be used for internal detect logic of the hold/reset logic of the LPM, or CDR <sup>(1)</sup> . The user needs to verify received data to decide whether or not an electrical idle state is present i.e., qualification using incoming data is essential in this mode of operation.
PCIe Gen2	See <a href="#">Figure 4-12</a> and <a href="#">Figure 4-13</a> for the algorithm to determine whether the RX is in electrical idle. While entering and exiting electrical idle, the EIOS detection must be used along with RXELECIDLE assertion to determine whether the RX is in electrical idle. RX electrical idle should not be used for internal detect logic of the hold/reset logic of the LPM, or CDR <sup>(1)</sup> . The user needs to verify received data to decide whether or not an electrical idle state is present i.e., qualification using incoming data is essential in this mode of operation.

**Notes:**

1. The attributes pertaining to LPM, and CDR are:

- RXCDR\_HOLD\_DURING\_EIDLE
- RXCDR\_FR\_RESET\_ON\_EIDLE
- RXCDR\_PH\_RESET\_ON\_EIDLE
- RX\_LPM\_HOLD\_DURING\_EIDLE
- RXBUF\_RESET\_ON\_EIDLE
- RXBUF\_EIDLE\_HI\_CNT
- RXBUF\_EIDLE\_LO\_CNT

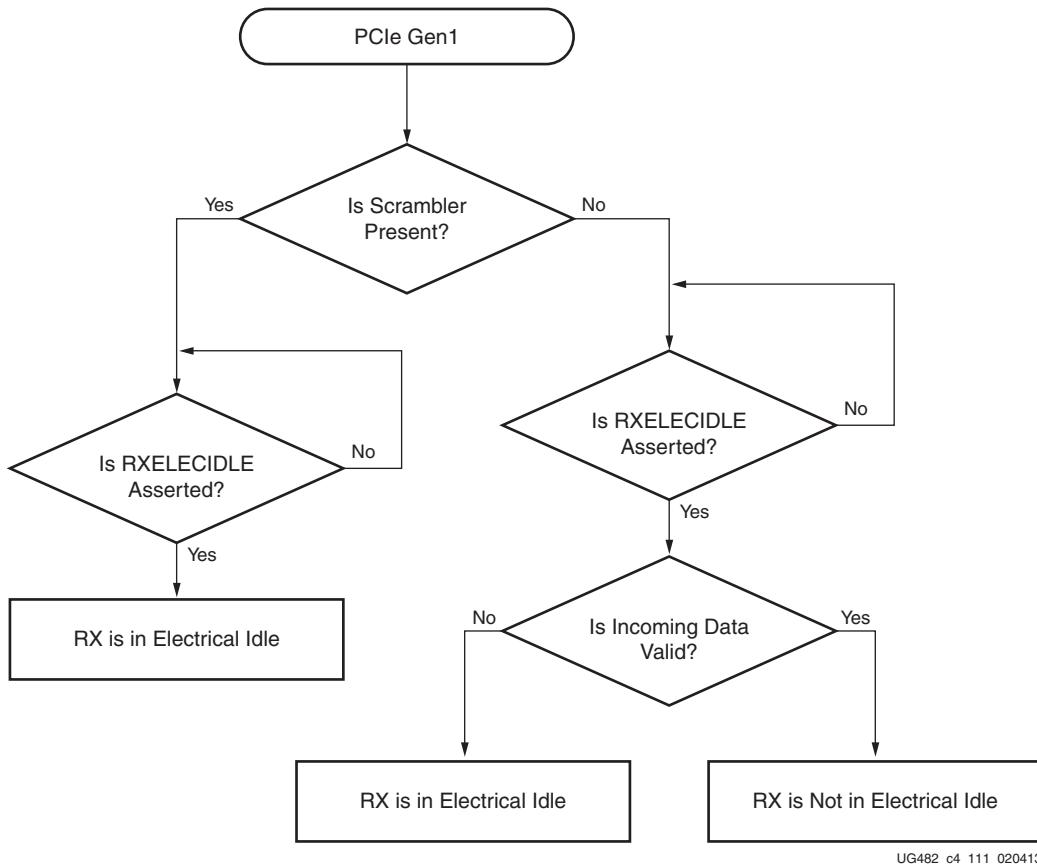


Figure 4-10: Flowchart for PCIe Gen1

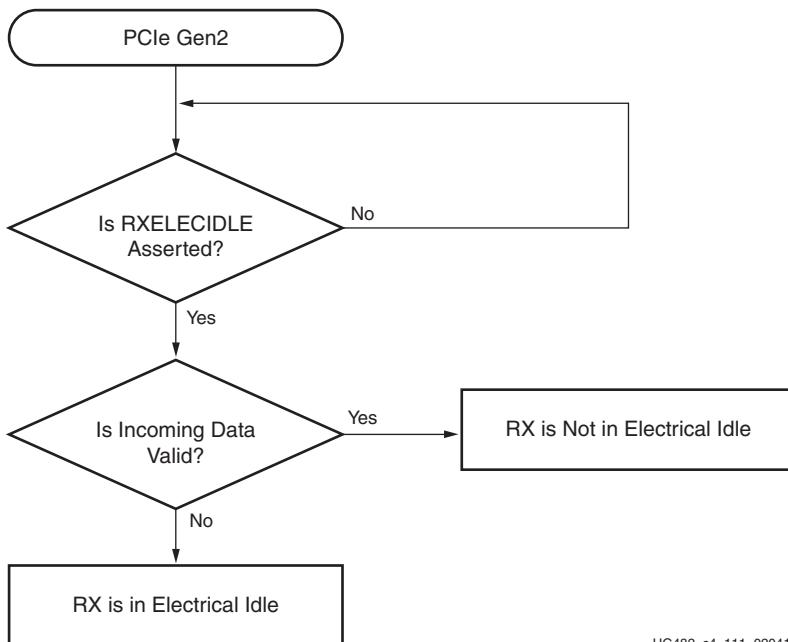
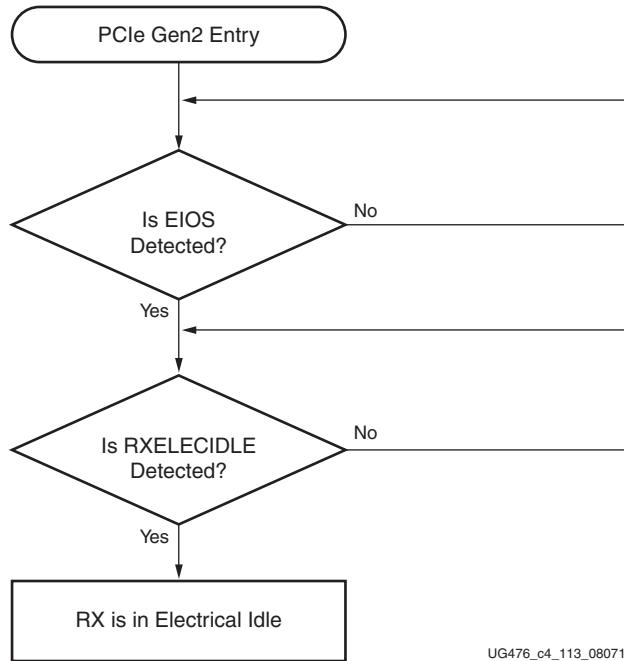
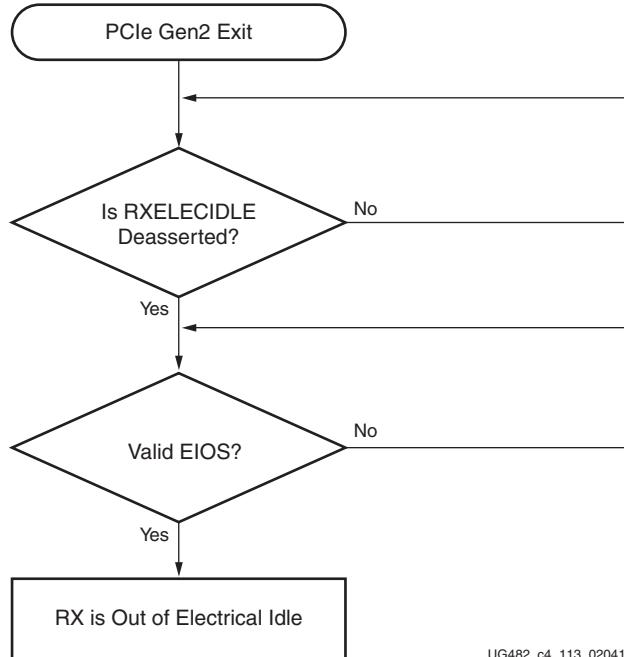


Figure 4-11: Flowchart for PCIe Gen2



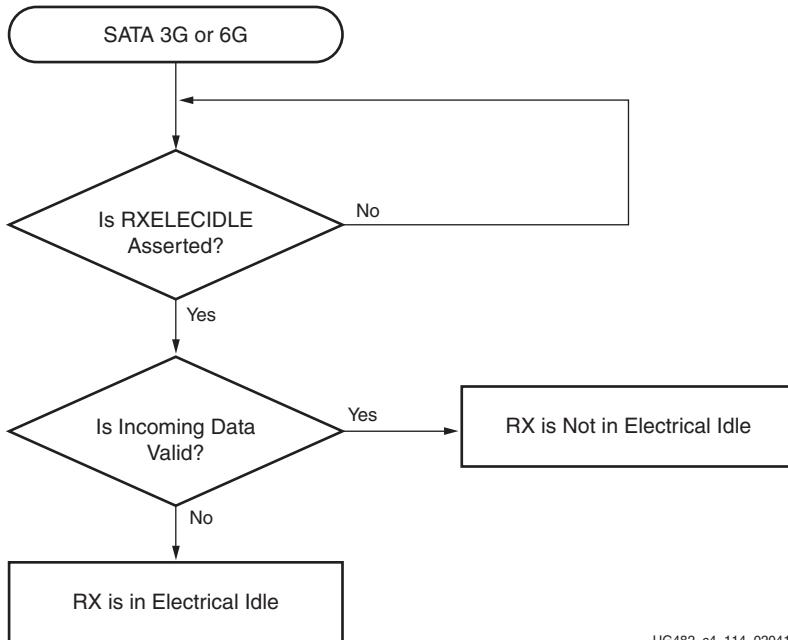
UG476\_c4\_113\_080712

Figure 4-12: Flowchart for Entry to RX Electrical Idle for PCIe Gen2



UG482\_c4\_113\_020413

Figure 4-13: Flowchart for Exit from RX Electrical Idle for PCIe Gen2



UG482\_c4\_114\_020413

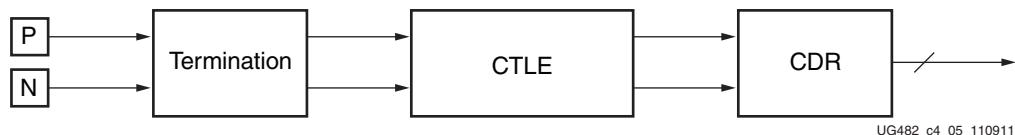
Figure 4-14: Flowchart for SATA 3G or SATA 6G

## RX Equalizer

### Functional Description

The GTP transceiver receiver has a power-efficient adaptive continuous time linear equalizer (CTLE) to compensate for signal distortion due to high-frequency attenuation in the physical channel. To maintain parity with the 7 series FPGAs GTx and GTH transceivers, the CTLE is referred to as the low-power mode (LPM).

The LPM mode (see Figure 4-15) has adaptive low and high frequency boosts. It is for applications with line rates up to 6.6 Gb/s for short reach applications, with channel losses of 12 dB or less at the Nyquist frequency.



UG482\_c4\_05\_110911

Figure 4-15: RX Equalization Block Diagram

### Ports and Attributes

Table 4-9 defines the RX equalizer ports.

Table 4-9: RX Equalizer Ports

Port	Dir	Clock Domain	Description
RXLPMRESET	In	Async	Resets the LPM circuitry.
RXLPMHFHOLD	In	Async	When set to 1'b1, the current value of the high-frequency boost is held. When set to 1'b0, the high-frequency boost is adapted.
RXLPMHFOVRDEN	In	Async	When set to 1'b1, the high-frequency boost is controlled by the RXLPM_HF_CFG attribute. When set to 1'b0, the high-frequency boost is controlled by the RXLPMHFHOLD signal.
RXLPMLFHOLD	In	Async	When set to 1'b1, the current value of the low-frequency boost is held. When set to 1'b0, the low-frequency boost is adapted.
RXLPMLFODREN	In	Async	When set to 1'b1, the low-frequency boost is controlled by the RXLPM_LF_CFG attribute. When set to 1'b0, the low-frequency boost is controlled by the RXLPMLFHOLD signal.

Table 4-10 defines the RX equalizer attributes.

Table 4-10: RX Equalizer Attributes

Attribute	Type	Description
ADAPT_CFG0	20-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RX_LPM_HOLD_DURING_EIDLE	1-bit Binary	When set to 1'b1, all adapted values for the LPM and offset cancellation are held when the GTP transceiver RX is in electrical idle and restored after electrical idle is exited.
RXLPMRESET_TIME	7-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXLPM_CFG	1-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

Table 4-10: RX Equalizer Attributes (Cont'd)

Attribute	Type	Description
RXLPM_CFG1	4-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXLPM_HF_CFG2	4-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXLPM_HF_CFG	14-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXLPM_BIAS_STARTUP_DISABLE	1-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXLPM_LF_CFG	18-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

## RX CDR

### Functional Description

The RX clock data recovery (CDR) circuit in each GTPE2\_CHANNEL transceiver extracts the recovered clock and data from an incoming data stream. [Figure 4-16](#) illustrates the architecture of the CDR block. Clock paths are shown with dotted lines for clarity.

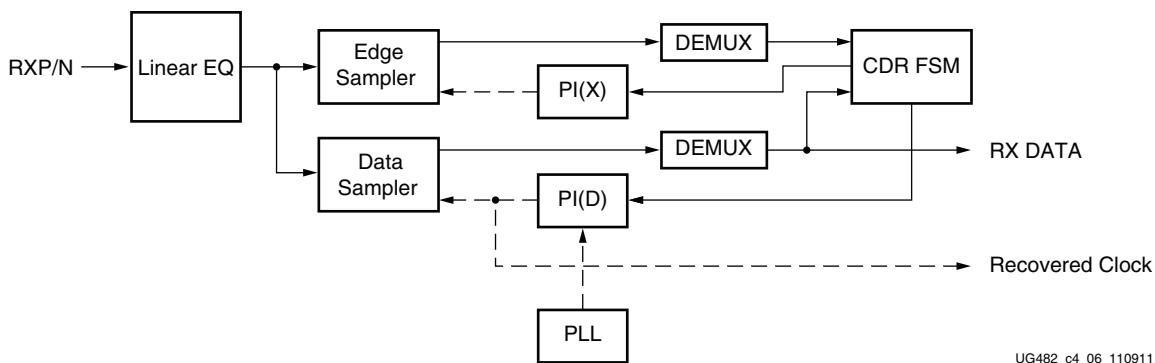
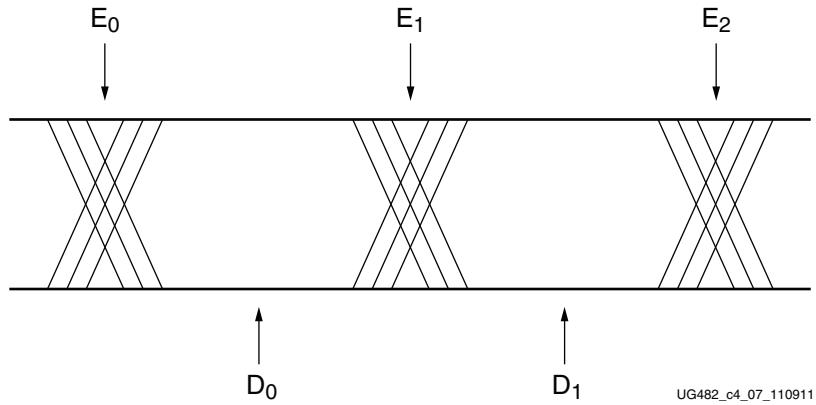


Figure 4-16: CDR Detail

The GTPE2\_CHANNEL transceiver employs phase rotator CDR architecture. Incoming data first goes through receiver equalization stages. The equalized data is captured by an edge and a data sampler. The data captured by the data sampler is fed to the CDR state machine and the downstream transceiver blocks.

The CDR state machine uses the data from both the edge and data samplers to determine the phase of the incoming data stream and to control the phase interpolators (PIs). The phase for the edge sampler is locked to the transition region of the data stream while the phase of the data sampler is positioned in the middle of the data eye.



**Figure 4-17: CDR Sampler Positions**

The PLL0 or PLL1 provides a base clock to the phase interpolator. The phase interpolator in turn produces fine, evenly spaced sampling phases to allow the CDR state machine to have fine phase control. The CDR state machine can track incoming data streams that can have a frequency offset from the local PLL reference clock.

## Ports and Attributes

[Table 4-11](#) defines the CDR ports.

**Table 4-11: CDR Ports**

Port	Dir	Clock Domain	Description
RXCDFREQRESET	In	Async	Reserved. Tied Low.
RXCDRHOLD	In	Async	Hold the CDR control loop frozen.
RXCDROVRDEN	In	Async	Reserved.
RXCDRRESET	In	Async	Reserved. Tied Low.
RXCDRRESETRSV	In	Async	Reserved.
RXRATE[2:0]	In	RXUSRCLK2 (RXRATEMODE makes this port asynchronous)	This port dynamically controls the setting for the RX serial clock divider D (see <a href="#">Table 4-16</a> ) and it is used with RXOUT_DIV attribute. 3'b000: Use RXOUT_DIV divider value 3'b001: Set D divider to 1 3'b010: Set D divider to 2 3'b011: Set D divider to 4 3'b100: Set D divider to 8 RXBUF_RESET_ON_RATE_CHANGE attribute enable optional automatic reset.
RXCDRLOCK	Out	Async	Reserved.

Table 4-11: CDR Ports (Cont'd)

Port	Dir	Clock Domain	Description
RXOSHOLD	In	Async	When set to 1'b1, the current value of the offset cancellation is held. When set to 1'b0, the offset cancellation is adapted.
RXOSOVRDEN	In	Async	When set to 1'b1, the Offset Cancellation is controlled by the RX_OS_CFG attribute. When set to 1'b0, the AGC is controlled by the RXSHOLD signal.
RXOSCALRESET	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTPD	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTCFG[3:0]	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used..
RXOSINTD0[3:0]	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTOVRDEN	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTSTROBE	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTHOLD	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTTESTOVRDEN	In	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTSTARTED	Out	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTSTROBESTARTED	Out	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXOSINTDONE	Out	Async	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

**Table 4-12** defines the CDR related attributes.

**Table 4-12: CDR Attributes**

Attribute	Type	Description
CFOK_CFG	43-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
CFOK_CFG2	7-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
CFOK_CFG3	7-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXCDR_CFG	83-bit Hex	CDR configuration. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXCDR_LOCK_CFG	6-bit Binary	CDR Lock loop configuration. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXCDR_HOLD_DURING_EIDLE	Binary	Enables the CDR to hold its internal states during an optional PCI Express reset sequence during electrical idle.
RXCDR_FR_RESET_ON_EIDLE	Binary	Enables automatic reset of the CDR frequency during the optional PCI Express reset sequence during electrical idle.
RXCDR_PH_RESET_ON_EIDLE	Binary	Enables automatic reset of the CDR phase during the optional PCI Express reset sequence during electrical idle.
RX_OS_CFG[12:0]	13-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.

## Use Modes

### RX CDR Lock to Reference

To get the CDR to lock to reference set RXCDRHOLD = 1'b1 and set RXCDROVRDEN = 1'b0

### Dynamically Changing RX CDR Settings for Line Rate and Selected Protocol Changes

The sequence of events to dynamically change the RX CDR settings is described here. It pertains only to changes for the CDR:

1. When ready (i.e., all valid data is flushed out of the receiver datapath), use the DRP to implement changes to the CDR loop filter settings with the attribute

RXCDR\_CFG[83:0]. Recommended settings for this attribute are provided in [Table 4-13](#), [Table 4-14](#), and [Table 4-15](#).

2. Provide the changes via ports PLL[0/1]REFCLKSEL and/or the DRP to the attributes listed in [Table 2-9, page 36](#).
3. Follow the reset guidelines as detailed in [PLL Reset, page 42](#).
4. When the PLL has locked, assert GTRXRESET and follow the guidelines detailed in [GTP Transceiver TX Reset in Response to GTTXRESET Pulse, page 46](#).
5. After the RXRESETDONE signal goes High, correct data must be verified before continuing with the operation of the transceiver (i.e., check a known data pattern).

## Dynamically Changing RX CDR Settings to Tune CDR Loop Filter Settings Only

1. When ready (all valid data flushed out of receiver datapath), use the DRP to implement changes to the CDR loop filter settings with the attribute RXCDR\_CFG[83:0]. Recommended settings for this attribute are provided in [Table 4-13](#), [Table 4-14](#), and [Table 4-15](#).
2. Assert the GTRXRESET port and follow the guidelines detailed in [GTP Transceiver RX Reset in Response to GTRXRESET Pulse, page 56](#).

After the RXRESETDONE signal goes High, correct data must be verified before continuing with the operation of the transceiver (i.e., check a known data pattern).

**Table 4-13: CDR Recommended Settings for Scrambled/PRBS Data<sup>(1)</sup> (No SSC<sup>(2)</sup>)**

RXOUT_DIV	REFCLK PPM	RXCDR_CFG
1	±200	83'h0_0011_07FE_2060_2104_1010
	±700	
	±1,250	
2	±200	83'h0_0011_07FE_2060_2108_1010
	±700	
	±1,250	
4 or 8	±200	83'h0_0011_07FE_0860_2110_1010
	±700	
	±1,250	

### Notes:

1. For protocol-specific settings, use the recommended value from the 7 Series FPGAs Transceivers Wizard and/or the protocol characterization reports.
2. Spread-spectrum clocking (SSC) is used to reduce the spectral density of electromagnetic interference (EMI).

Table 4-14: CDR Recommended Settings for Protocols with SSC

RXOUT_DIV	REFCLK PPM with SSC	RXCDR_CFG
1	$\pm 700$ PPM SSC 33 KHz Triangular -5,000 PPM	83'h0_0000_87FE_2060_2448_1010
2		83'h0_0000_47FE_2060_2450_1010
4		83'h0_0000_47FE_1060_2450_1010

Table 4-15: GTP CDR Recommended Settings for 8B/10B Encoded Data<sup>(1)</sup> (No SSC<sup>(2)</sup>)

RXOUT_DIV	REFCLK PPM	RXCDR_CFG
1	$\pm 200$	83'h0_0001_07FE_4060_0104_1010
	$\pm 700$	83'h0_0001_07FE_4060_2104_1010
	$\pm 1,250$	
2	$\pm 200$	83'h0_0001_07FE_2060_0104_1010
	$\pm 700$	83'h0_0001_07FE_2060_2104_1010
	$\pm 1,250$	
4	$\pm 200$	83'h0_0001_07FE_1060_0104_1010
	$\pm 700$	83'h0_0001_07FE_1060_2104_1010
	$\pm 1,250$	
8	$\pm 200$	83'h0_0001_07FE_0860_0104_1010
	$\pm 700$	83'h0_0001_07FE_0860_2104_1010
	$\pm 1,250$	

**Notes:**

- For protocol-specific settings, use the recommended value from the 7 Series FPGAs Transceivers Wizard and/or the protocol characterization reports.
- Spread-spectrum clocking (SSC) is used to reduce the spectral density of electromagnetic interference (EMI).
- RX\_DEBUG\_CFG is 14'h000 for all settings.

# RX Fabric Clock Output Control

## Functional Description

The RX clock divider control block has two main components: serial clock divider control and parallel clock divider and selector control. The clock divider and selector details are illustrated in [Figure 4-18](#).

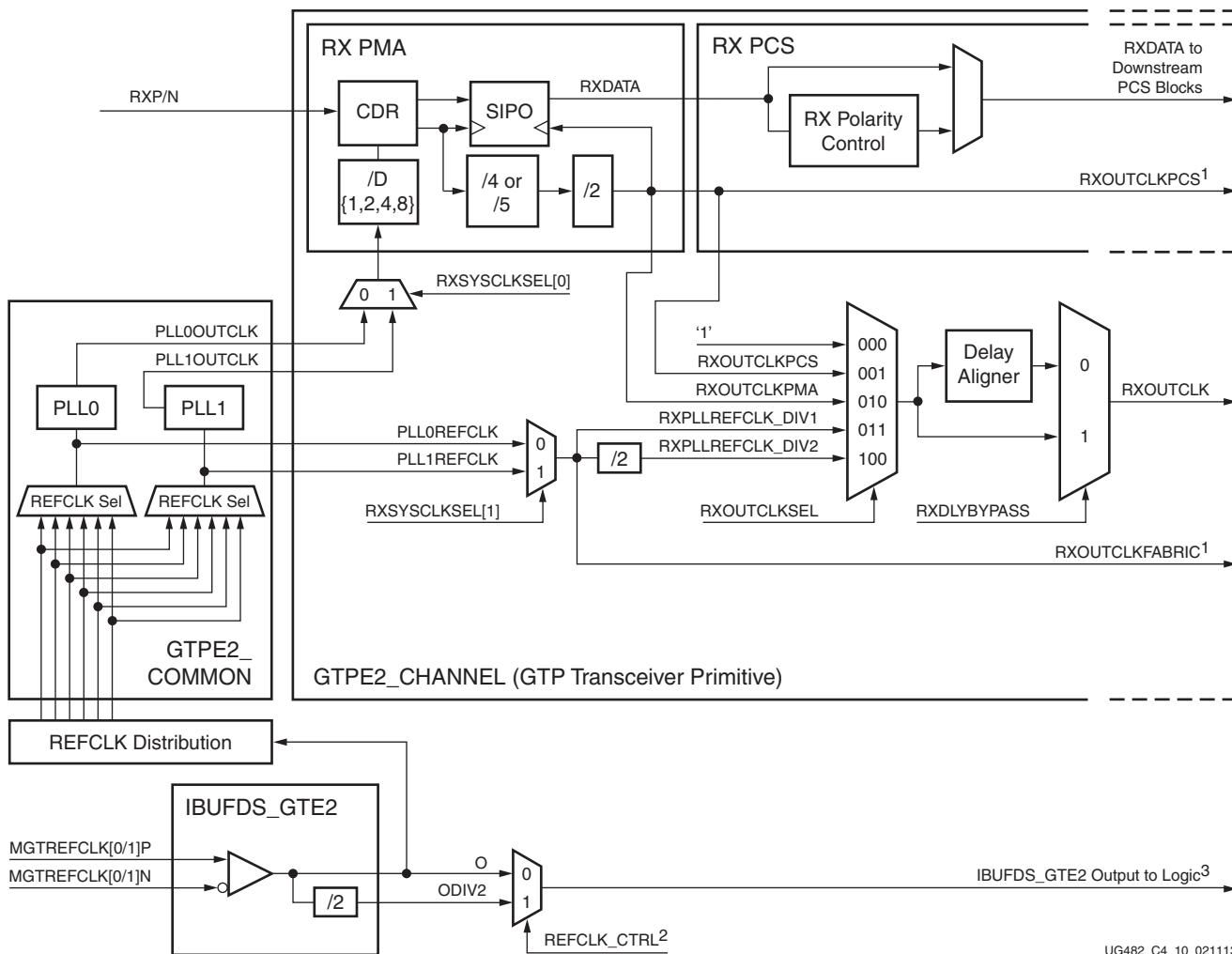


Figure 4-18: RX Serial and Parallel Clock Divider

Note relevant to [Figure 4-18](#):

1. RXOUTCLKPCS and RXOUTCLKFABRIC are redundant outputs. RXOUTCLK should be used for new designs.
2. The REFCLK\_CTRL option is controlled automatically by software and is not user selectable. The user can only route one of IBUFDS\_GTE2's O or ODIV2 outputs to the FPGA logic.
3. IBUFDS\_GTE2 is a redundant output for additional clocking scheme flexibility.

4. The selection of the /4 or /5 divider block is controlled by the RX\_DATA\_WIDTH attribute from the GTPE2\_CHANNEL primitive. /4 is selected when RX\_DATA\_WIDTH = 16 or 32. /5 is selected when RX\_DATA\_WIDTH = 20 or 40.
5. For details about placement constraints and restrictions on clocking resources (MMCME2, PLLE2, IBUFDS\_GTE2, BUFG, etc.), refer to [UG472, 7 Series FPGAs Clocking Resources User Guide](#).

## Serial Clock Divider

Each transmitter PMA module has a D divider that divides down the clock from the PLL for lower line rate support. This serial clock divider, D, can be set statically for applications with a fixed line rate or it can be changed dynamically for protocols with multiple line rates. The control for the serial divider is described in [Table 4-16](#). For details about the line rate range per speed grade, refer to the [7 series FPGAs documentation page](#) for the appropriate data sheet.

To use the D divider in fixed line rate applications, the RXOUT\_DIV attribute must be set to the appropriate value, and the RXRATE port needs to be tied to 3'b000. Refer to the Static Setting via Attribute column in [Table 4-16](#) for details.

To use the D divider in multiple line rate applications, the RXRATE port is used to dynamically select the D divider value. The RXOUT\_DIV attribute and the RXRATE port must select the same D divider value upon device configuration. After device configuration, the RXRATE is used to dynamically change the D divider value. Refer to the Dynamic Control via Ports column in [Table 4-16](#) for details.

**Table 4-16: RX PLL Output Divider Setting**

D Divider Value	Static Setting via Attribute	Dynamic Control via Ports
1	RXOUT_DIV = 1 RXRATE = 3'b000	RXOUT_DIV = Ignored RXRATE = 3'b001
2	RXOUT_DIV = 2 RXRATE = 3'b000	RXOUT_DIV = Ignored RXRATE = 3'b010
4	RXOUT_DIV = 4 RXRATE = 3'b000	RXOUT_DIV = Ignored RXRATE = 3'b011
8	RXOUT_DIV = 8 RXRATE = 3'b000	RXOUT_DIV = Ignored RXRATE = 3'b100

## Parallel Clock Divider and Selector

The parallel clock outputs from the RX clock divider control block can be used as a fabric logic clock depending on the line rate and protocol requirements.

The recommended clock for the FPGA logic is the RXOUTCLK from one of the GTP transceivers. It is also possible to bring the MGTREFCLK directly to the fabric and use as the fabric clock. RXOUTCLK is preferred for general applications because it has an output delay control used for applications that bypass the RX buffer for constant datapath delay. Refer to [RX Buffer Bypass, page 177](#) for more details.

The RXOUTCLKSEL port controls the input selector and allows these clocks to be output via RXOUTCLK port:

- RXOUTCLKSEL = 3'b001: RXOUTCLKPCS path is not recommended to be used as it incurs extra delay from the PCS block.

- RXOUTCLKSEL = 3'b010: RXOUTCLKPMA is the recovered clock that can be brought out to the FPGA logic. The recovered clock is used by protocols that do not have a clock compensation mechanism and require to use a clock synchronous to the data (the recovered clock), to clock the downstream fabric logic. It is also used by the RX PCS block. This clock is interrupted when the PLL or CDR is reset by one of the related reset signals.
- RXOUTCLKSEL = 3'b011 or 3'b100: RXPLLREFCLK\_DIV1 or RXPLLREFCLK\_DIV2 is the input reference clock to the PLL0 or PLL1 depending on the RXSYSCLKSEL[1] setting. For usages that do not require outputting a recovered clock to the fabric, RXPLLREFCLK\_DIV1 or RXPLLREFCLK\_DIV2 can be used as the system clock. However, TXOUTCLK is usually used as system clock.

## Ports and Attributes

[Table 4-17](#) defines the ports required for RX fabric clock output control.

**Table 4-17: RX Fabric Clock Output Control Ports**

Port	Dir	Clock Domain	Description
RXOUTCLKSEL[2:0]	In	Async	This port controls the multiplexer select signal in <a href="#">Figure 4-18</a> . 3'b000: Static 1 3'b001: RXOUTCLKPCS path 3'b010: RXOUTCLKPMA path 3'b011: RXPLLREFCLK_DIV1 path 3'b100: RXPLLREFCLK_DIV2 path Others: Reserved.
RXRATE[2:0]	In	RXUSRCLK2 (RXRATEMODE makes this port asynchronous)	This port dynamically controls the setting for the RX serial clock divider D (see <a href="#">Table 4-16</a> ) and it is used with RXOUT_DIV attribute. 3'b000: Use RXOUT_DIV divider value 3'b001: Set D divider to 1 3'b010: Set D divider to 2 3'b011: Set D divider to 4 3'b100: Set D divider to 8
RXOUTCLKFABRIC	Out	Clock	RXOUTCLKFABRIC is a redundant output reserved for testing. RXOUTCLK with RXOUTCLKSEL = 3'b011 should be used instead.
RXOUTCLK	Out	Clock	RXOUTCLK is the recommended clock output to the FPGA logic. The RXOUTCLKSEL port is the input selector for RXOUTCLK and allows the PLL input reference clock to the FPGA logic.
RXOUTCLKPCS	Out	Clock	RXOUTCLKPCS is a redundant output. RXOUTCLK with RXOUTCLKSEL = 3'b001 should be used instead.
RXRATEDONE	Out	RXUSRCLK2	The RXRATEDONE port is asserted High for one RXUSRCLK2 cycle in response to a change on the RXRATE port. The TRANS_TIME_RATE attribute defines the period of time between a change on the RXRATE port and the assertion of RXRATEDONE.

Table 4-17: RX Fabric Clock Output Control Ports (Cont'd)

Port	Dir	Clock Domain	Description
RXRATEMODE	In	Async	Determines if RXRATE should be treated as synchronous or asynchronous. 0: Synchronous. When set to 1'b0, an automatic reset sequence occurs in response to a change on the RXRATE port. 1: Asynchronous.
RXDLYBYPASS	In	Async	RX delay alignment bypass: 0: Uses the RX delay alignment circuit. Set to 1'b0 when the RX buffer is bypassed. 1: Bypasses the RX delay alignment circuit. Set to 1'b1 when the RX buffer is used.

Table 4-18 defines the attributes required for RX fabric clock output control.

Table 4-18: RX Fabric Clock Output Control Attributes

Attribute	Type	Description
TRANS_TIME_RATE	8-bit Hex	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used. This attribute determines when PHYSTATUS and RXRATEDONE are asserted after a rate change.
RXBUFF_RESET_ON_RATE_CHANGE	String	When set to TRUE, this attribute enables automatic RX buffer reset during a rate change event initiated by a change in RXRATE.
RXOUT_DIV	Integer	This attribute controls the setting for the RX serial clock divider. This attribute is only valid when RXRATE = 3'b000. Otherwise the D divider value is controlled by RXRATE. Valid settings are 1, 2, 4, and 8.

## Using RXRATE

When users want to change the divider D setting via RXRATE, the steps in Figure 4-19 should be performed:

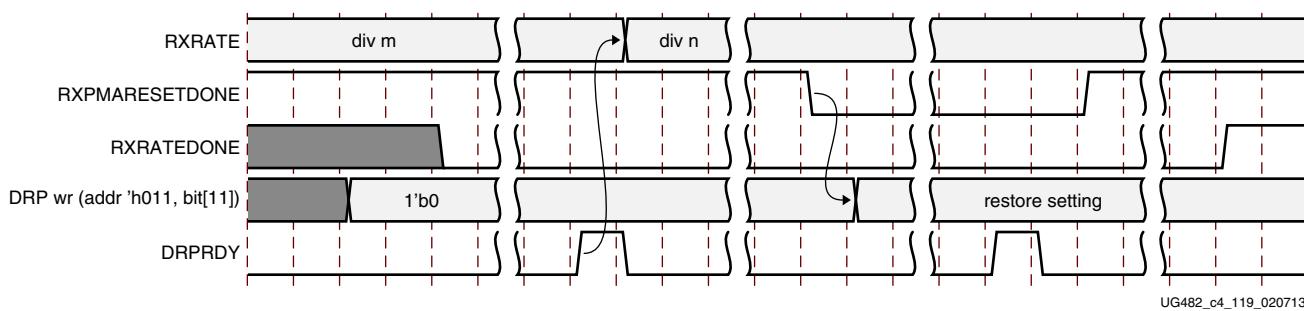


Figure 4-19: RXRATE change example

Notes relevant to Figure 4-19:

1. DRP wr denotes the function of performing a DRP write to Addr 9'h011. The exact DRP transaction is not shown.

2. The sequence of events in Figure 4-19 is not drawn to scale.
3. To change RXRATE, a DRP write must be issued to the GTPE2\_CHANNEL primitive, DRPADDR 9'h011, set bit[11] to 1'b0.
  - a. To ensure only bit[11] of DRPADDR 9'h011 is modified, a read-modify-write function must be performed.
4. Upon DRP write completion, the value of RXRATE must be changed to the new desired setting.
5. Wait for RXPMARESETDONE to be detected low.
6. Issue DRP write to the GTPE2\_CHANNEL primitive, DRPADDR 9'h011, restoring the original setting for bit[11]. The completion of this DRP write must occur before RXPMARESETDONE switches from low to high. RXPMARESETDONE will stay low for a minimum of 0.66  $\mu$ s.
7. This sequence will simulate correctly if SIM\_RESET\_SPEEDUP is set to FALSE. If SIM\_RESET\_SPEEDUP is set to TRUE, this sequence should be bypassed.

## RX Margin Analysis

### Functional Description

As line rates and channel attenuation increase, the receiver equalizers are more often enabled to overcome channel attenuation. This poses a challenge to system bring-up because the quality of the link cannot be determined by measuring the far-end eye opening at the receiver pins. At high line rates, the received eye measured on the printed circuit board can appear to be completely closed even though the internal eye after the receiver equalizer is open.

The 7 series FPGAs GTP transceivers RX eye scan provides a mechanism to measure and visualize the receiver eye margin after the equalizer. Additional use modes enable several other methods to determine and diagnose the effects of equalization settings.

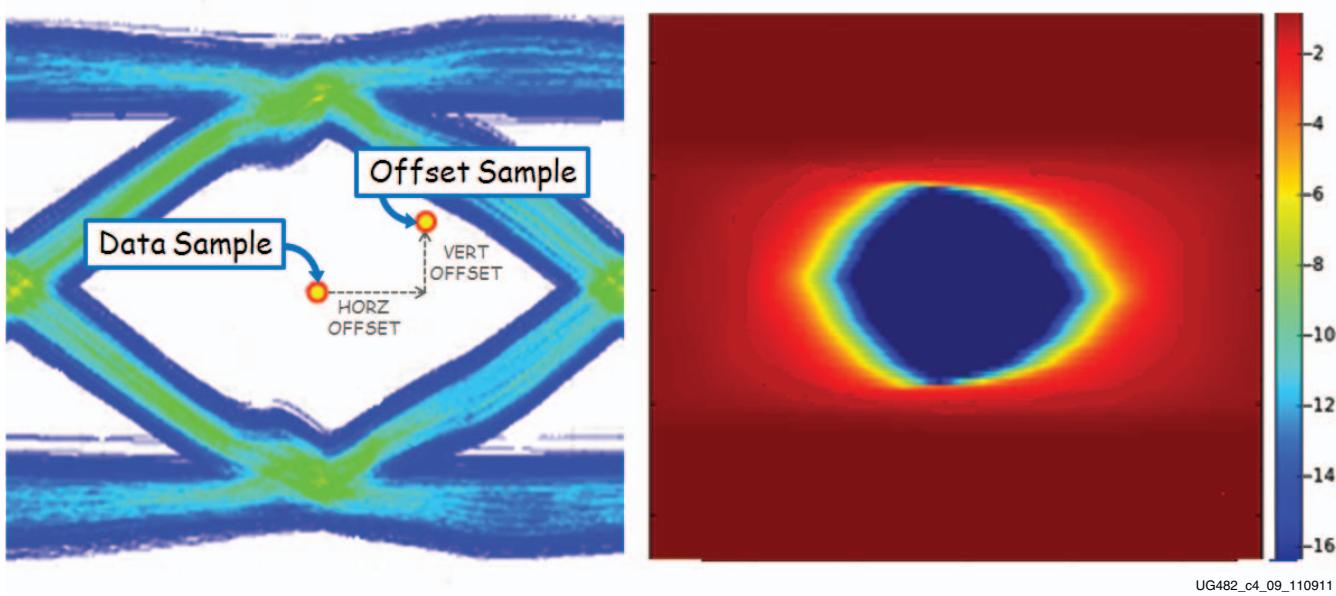


Figure 4-20: Offset Sample and Data Sample to Calculate BER as a Function of Offset—the Statistical Eye

## Eye Scan Theory

RXDATA is recovered from the equalized differential waveform by sampling after the RX equalizer. The horizontal sampling position is determined by the CDR function and the vertical position is differential zero. This is indicated as data sample in [Figure 4-20](#).

To enable eye scan functionality, an additional sampler is provided with programmable (horizontal and vertical) offsets from the data sample point. This is indicated as offset sample in [Figure 4-20](#).

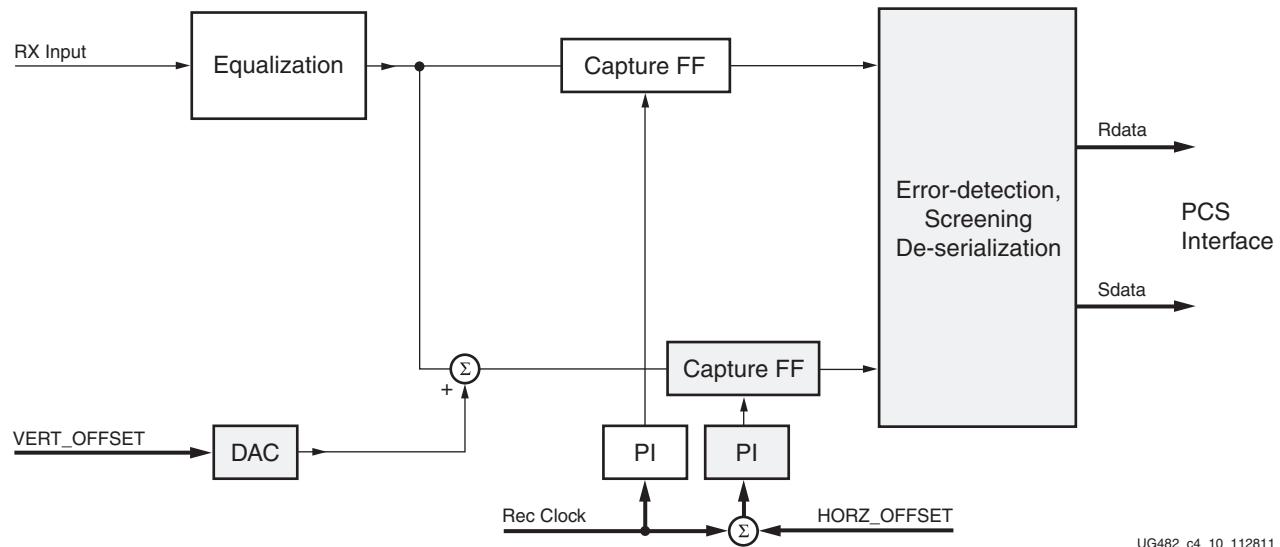
A single eye scan measurement consists of accumulating the number of data samples (sample count) and the number of times that the offset sample disagreed with the data sample (error count). The bit error ratio (BER) at the programmed vertical and horizontal offset is the ratio of the error count to the sample count. The sample count can range from tens of thousands to greater than  $10^{14}$ .

Repeating such BER measurements for the full array of horizontal and vertical offsets (or a subsampled set of offsets) produces a BER map as shown in [Figure 4-20](#), commonly referred to as a *statistical eye*, where the color map represents  $\log_{10}(\text{BER})$ . In this view, the eye is apparently smaller than a traditional oscilloscope view (as in [Figure 4-20](#)) because it has been closed by very low probability jitter and noise that does not show up in the much lower number of samples of an oscilloscope.

Because this functionality puts no restrictions on the data patterns being received nor requires any changes in the RX settings, it can be performed while application data is being received without error. Furthermore, no FPGA logic is required—only the ability to read and write attributes.

## Eye Scan Architecture

The blocks with shaded gray in [Figure 4-21](#) describe the portion of the PMA architecture that supports eye scan. The horizontal offset (HORZ\_OFFSET) advances or delays the sampling time of the offset samples relative to the data samples. The vertical offset (VERT\_OFFSET) raises or lowers the differential voltage threshold to which the equalized waveform is compared. The data samples are deserialized into the Rdata bus, and the offset samples are deserialized into the Sdata bus.



UG482\_c4\_10\_112811

Figure 4-21: PMA Architecture to Support Eye Scan

Figure 4-22 describes the portion of the PCS architecture that supports eye scan. The 40-bit Rdata bus contains the data samples, and each bit of the 40-bit Sdata bus is one if and only if the corresponding data sample and offset sample are not equal. (See ES\_ERRDET\_EN in Table 4-20, page 158.)

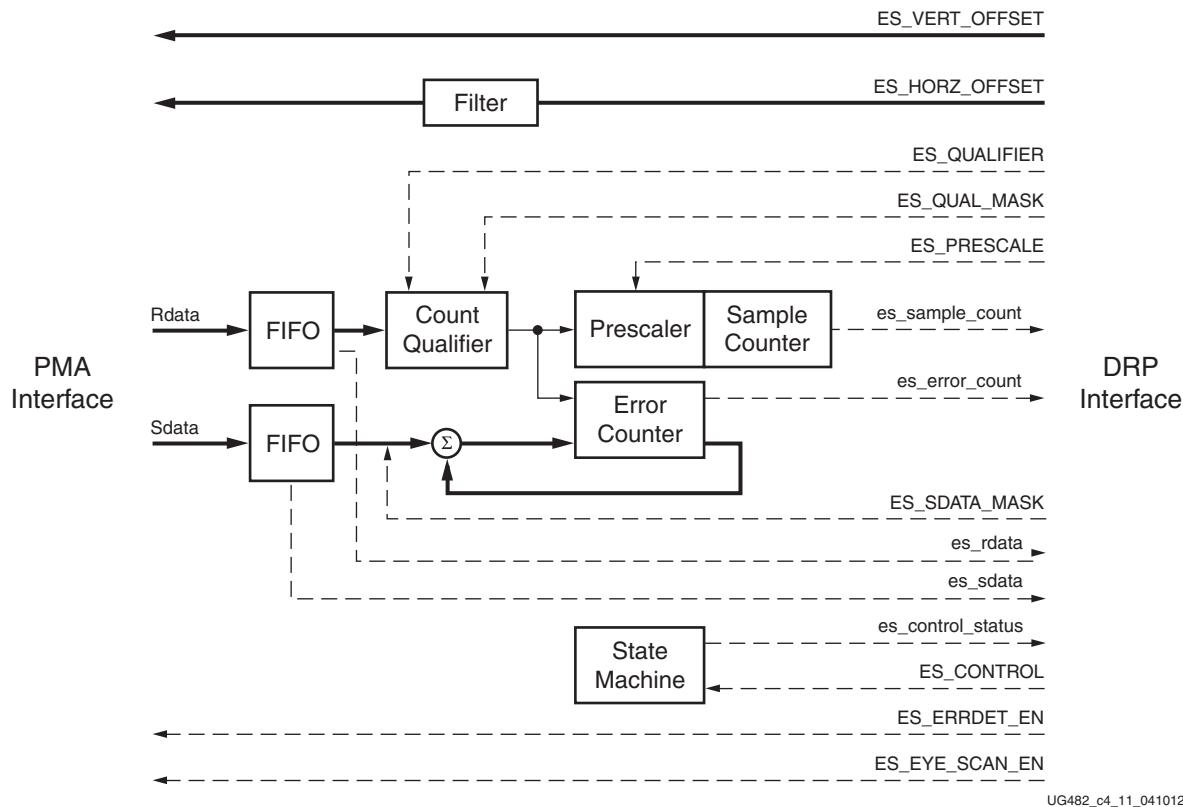


Figure 4-22: PCS Architecture to Support Eye Scan

Two consecutive cycles of Sdata are masked by ES\_SDATA\_MASK[79:0] (i.e., bit-by-bit Sdata[i] AND NOT mask[i]). The algebraic sum of bits [39:0] of this result is the number of errors to be added in the error counter.

Two consecutive cycles of Rdata are compared with the pattern in ES\_QUALIFIER[79:0], and that result is masked by (i.e., bit-by-bit ORed with) ES\_QUAL\_MASK[79:0]. The logical AND of this result determines whether the prescaler/sample counter is incremented and the errors added to the error counter. For a statistical eye, ES\_QUAL\_MASK is 80'b1, so the sample counter and error counter accumulate on every cycle. ES\_SDATA\_MASK unmasks only the current data (bit 39 and below; see the description of RX\_DATA\_WIDTH) to avoid double counting errors because they appear first in the lower 40 bits and then in the upper 40 bits on the next cycle.

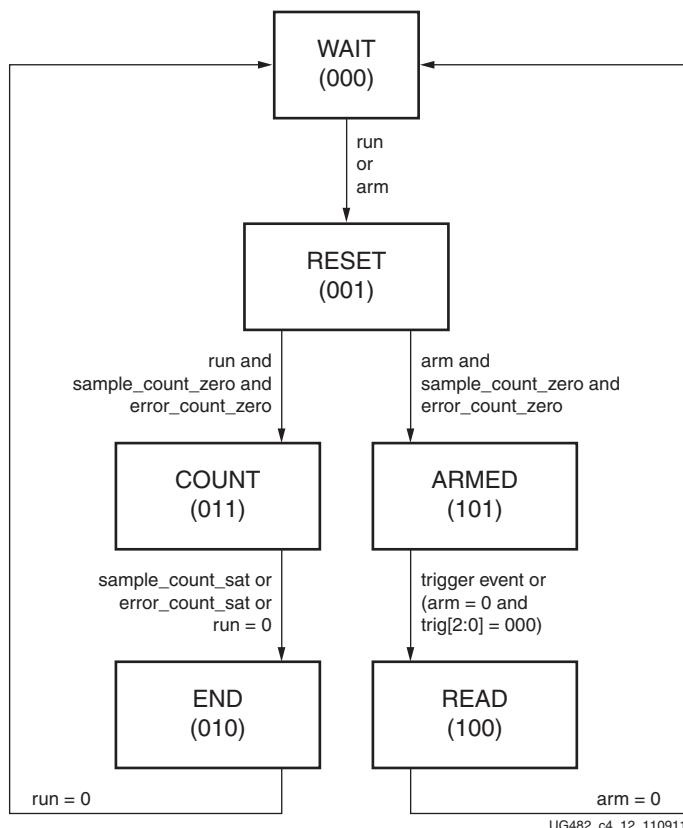
Alternate use modes produce scope-like displays by unmasking a sequence of Rdata bits (up to 20), causing error and sample accumulation only if Rdata matches ES\_QUALIFIER in that range of bits. In these use modes, only one Sdata bit per measurement is unmasked. In diagnostic use modes, Rdata and Sdata are *frozen* and can be read out through the DRP interface when:

- An error occurs,
- A count qualifier occurs,
- A fabric port causes a trigger, or

- A trigger is forced via an attribute write.

The diagnostic use modes could be used, for example, to examine the pattern of burst errors due to equalization behavior.

**Figure 4-23** documents the state transitions in the eye scan state machine.



**Figure 4-23: Eye Scan State Machine**

ES\_CONTROL[1:0] are the signals arm and run, respectively. From the WAIT state, run initiates the BER measurement loop (left) and arm starts the diagnostic loop (right).

The RESET state zeros the error and sample counters, then enters the COUNT state or the ARMED state (depending on whether run or arm is active).

In the COUNT state, samples and errors are accumulated in the counters. When either counter is saturated, both counters stop and transition to the END state. This transition to the END state is detected by polling es\_control\_status[3:0]. Bit 0 (done) is set active only in the END, READ, and WAIT states. Bits [3:1] display the current state of the state machine.

The END state transitions to the WAIT state when run is set back to zero. The es\_sample\_count[15:0] and es\_error\_count[15:0] can be read either in the END or WAIT state.

In the ARMED state, the FIFOs (successive cycles of Rdata and of Sdata) are stopped when a trigger event occurs. The trigger event is either the count qualifier pulse, the logical OR of all bits into the error counter, or a manual trigger provided from a DRP data input or from a port. One of these four options is selected by trig[3:0] = ES\_CONTROL[5:2].

In the READ state, the last two cycles of Rdata can be read from the COE status register, es\_rdata[79:0], and the last two cycles of Sdata can be read from the COE status register, es\_sdata[79:0].

## Ports and Attributes

**Table 4-19** defines ports related to the RX eye scan function.

**Table 4-19: RX Margin Analysis Ports**

Port	Dir	Domain	Description
EYESCANDATAERROR	Out	async	Asserts high for one REC_CLK cycle when an (unmasked) error occurs while in the COUNT or ARMED state.
EYESCANTRIGGER	In	RXUSRCLK2	Causes a trigger event. See ES_CONTROL[4] below.
RXRATE	In	RXUSRCLK2	This port dynamically controls the setting for the RX serial clock divider D (see <b>Table 4-16</b> ) and it is used with RXOUT_DIV attribute.  3'b000: Use RXOUT_DIV divider value 3'b001: Set D divider to 1 3'b010: Set D divider to 2 3'b011: Set D divider to 4 3'b100: Set D divider to 8

**Table 4-20** defines RX eye scan attributes. Lower case attribute names indicate R/O.

**Table 4-20: RX Margin Analysis Attributes**

Attribute	Type	Description																				
ES_VERT_OFFSET	9-bit Binary	Controls the vertical (differential voltage) offset of the scan sample: [6:0]: Offset magnitude. [7]: Offset sign (1 is negative, 0 is positive).																				
ES_HORZ_OFFSET	12-bit Hex	Controls the horizontal (phase) offset of the scan sample. [10:0]: Phase offset (two's complement). The center of data eye (0 UI) corresponds to a count of 11'b0 for all data rates. The table below lists the minimum count (representing -0.5 UI) and maximum count (representing +0.5 UI) for each data rate.  <table border="0"> <tr> <th>Rate</th> <th>min_count [dec(bin)]</th> <th>eye_center [dec(bin)]</th> <th>max_count [dec(bin)]</th> </tr> <tr> <td>Full</td> <td>-32 (11'b11111100000)</td> <td>+0 (11'b00000000000)</td> <td>+32 (11'b00000100000)</td> </tr> <tr> <td>Half</td> <td>-64 (11'b11111000000)</td> <td>+0 (11'b00000000000)</td> <td>+64 (11'b00001000000)</td> </tr> <tr> <td>Qtr</td> <td>-128 (11'b11110000000)</td> <td>+0 (11'b00000000000)</td> <td>+128 (11'b00010000000)</td> </tr> <tr> <td>Octal</td> <td>-256 (11'b11100000000)</td> <td>+0 (11'b00000000000)</td> <td>+256 (11'b00100000000)</td> </tr> </table> [11]: Phase unification. Must be set to 0 for all positive counts (including zero) and to 1 for all negative counts.	Rate	min_count [dec(bin)]	eye_center [dec(bin)]	max_count [dec(bin)]	Full	-32 (11'b11111100000)	+0 (11'b00000000000)	+32 (11'b00000100000)	Half	-64 (11'b11111000000)	+0 (11'b00000000000)	+64 (11'b00001000000)	Qtr	-128 (11'b11110000000)	+0 (11'b00000000000)	+128 (11'b00010000000)	Octal	-256 (11'b11100000000)	+0 (11'b00000000000)	+256 (11'b00100000000)
Rate	min_count [dec(bin)]	eye_center [dec(bin)]	max_count [dec(bin)]																			
Full	-32 (11'b11111100000)	+0 (11'b00000000000)	+32 (11'b00000100000)																			
Half	-64 (11'b11111000000)	+0 (11'b00000000000)	+64 (11'b00001000000)																			
Qtr	-128 (11'b11110000000)	+0 (11'b00000000000)	+128 (11'b00010000000)																			
Octal	-256 (11'b11100000000)	+0 (11'b00000000000)	+256 (11'b00100000000)																			
ES_PRESCALE	5-bit Binary	Controls the pre-scaling of the sample count to keep both sample count and error count in reasonable precision within the 16-bit register range. Prescale = $2^{(1 + \text{register value})}$ , so minimum prescale is $2^{(1+0)} = 2$ and maximum prescale is $2^{(1+31)} = 4,284,967,296$ .																				

Table 4-20: RX Margin Analysis Attributes (*Cont'd*)

Attribute	Type	Description
ES_SDATA_MASK	80-bit Hex	<p>This attribute masks up to two cycles of the 40-bit Sdata bus. Binary 1 causes the corresponding bus bit to be masked and binary 0 leaves it unmasked. To support the statistical eye view, the error counter accumulates the total number of unmasked 1s on the most recent cycle of the Sdata bus (masked by ES_SDATA_MASK[39:0]). To support the scope and waveform views, the error counter increments by only one for any non-zero number of unmasked 1s on the previous cycle of the Sdata bus (masked by ES_SDATA_MASK[79:40]).</p> <p>This attribute and ES_QUAL_MASK must also mask out unused bits. For the statistical eye view, this attribute would assume the following values as a function of bus width:</p> <ul style="list-style-type: none"> <li>20-bit width: ES_SDATA_MASK = (40'b1, 20'b0, 20'b1)</li> <li>16-bit width: ES_SDATA_MASK = (40'b1, 16'b0, 24'b1)</li> </ul> <p>Scope and waveform views require a sequence of measurements, unmasking only a single bit per measurement.</p>
ES_QUALIFIER	80-bit Hex	<p>Eye scan can qualify BER measurements based on patterns up to 20 contiguous bits long in any position in the input data. Because the data, and therefore the qualifier pattern, is not aligned, the position of the pattern must be discovered by a barrel-shifting search. For example, looking for the pattern 10'b0011111010 (K28.5 in 8B/10B code) with a 20-bit data width would require a sequence of measurements such as the following, searching for a non-zero sample count at the correct alignment:</p> <ul style="list-style-type: none"> <li>ES_QUALIFIER = (50'b?, 10'b0011111010, 20'b?)</li> <li>ES_QUALIFIER = (49'b?, 10'b0011111010, 21'b?)</li> <li>ES_QUALIFIER = (48'b?, 10'b0011111010, 22'b?)</li> <li>...etc... (where ? represents a DON'T CARE bit that will be masked)</li> </ul> <p>The qualifier pattern is shifted only over the valid bits for the bus width (40, 32, 20, or 16).</p>
ES_QUAL_MASK	80-bit Hex	<p>This attribute masks those bits not included in the qualifier pattern. For example, the corresponding values for the K28.5 example above would be:</p> <ul style="list-style-type: none"> <li>ES_QUAL_MASK = (50'b1, 10'b0, 20'b1)</li> <li>ES_QUAL_MASK = (49'b1, 10'b0, 21'b1)</li> <li>ES_QUAL_MASK = (48'b1, 10'b0, 22'b1)</li> <li>...etc...</li> </ul>
ES_EYE_SCAN_EN	1-bit Binary	<p>This bit should always be 1 when using eye scan. Setting this bit to 0 powers down the eye scan circuitry in the PMA and forces the eye scan state to WAIT. Re-enabling eye scan functionality requires reasserting this bit and asserting/deasserting PMA reset.</p>
ES_ERRDET_EN	1-bit Binary	<p>1: Each bit of the Sdata bus is 1 if and only if the corresponding offset data sample does not agree with the recovered data sample. This is used for the statistical eye view.</p> <p>0: Each bit of the Sdata bus is the recovered data sample. Therefore, if no errors occurred, the Sdata bus would be identical to the Rdata bus. This is used for the scope and waveform views.</p>

Table 4-20: RX Margin Analysis Attributes (*Cont'd*)

Attribute	Type	Description												
ES_CONTROL	6-bit Binary	[0]: RUN. Asserting this bit causes a state transition from the WAIT state to the RESET state, initiating a BER measurement sequence. [1]: ARM Asserting this bit causes a state transition from the WAIT state to the RESET state, initiating a diagnostic sequence. In the ARMED state, deasserting this bit causes a state transition to the READ state if one of the states of bits [5:2] below is not met. [5:2]: 0001 In the ARMED state, causes a trigger event (transition to the READ state) when an error is detected (i.e., an unmasked 1 on the Sdata bus). 0010 In the ARMED state, causes a trigger event (transition to the READ state) when the qualifier pattern is detected in Rdata. 0100 In the ARMED state, causes a trigger event (transition to the READ state) when the eye_scan_trigger port asserts High. 1000 In the ARMED state, causes a trigger event (transition to the READ state) immediately.												
es_control_status	4-bit Binary	[0]: DONE. Asserted High only in the WAIT, END, or READ states. [3:1]: Current state of the state machine: <table style="margin-left: 20px;"> <tr><td>WAIT</td><td>000</td></tr> <tr><td>RESET</td><td>001</td></tr> <tr><td>COUN</td><td>011</td></tr> <tr><td>END</td><td>010</td></tr> <tr><td>ARMED</td><td>101</td></tr> <tr><td>READ</td><td>100</td></tr> </table>	WAIT	000	RESET	001	COUN	011	END	010	ARMED	101	READ	100
WAIT	000													
RESET	001													
COUN	011													
END	010													
ARMED	101													
READ	100													
es_rdata	80-bit Binary	When a trigger event occurs in the ARMED state, es_rdata[39:0] is the present state of the Rdata bus and es_rdata[79:40] is the previous state of the Rdata bus.												
es_sdata	80-bit Binary	When a trigger event occurs in the ARMED state, es_sdata[39:0] is the present state of the Sdata bus and es_sdata[79:40] is the previous state of the Sdata bus.												
es_error_count	16-bit Hex	In END and WAIT states, contains the final error count for the preceding BER measurement.												
es_sample_count	16-bit Hex	In END and WAIT states, contains the final sample count for the preceding BER measurement.												
RX_DATA_WIDTH	Integer	Sets the bit width of the RXDATA port. When 8B/10B encoding is enabled, RX_DATA_WIDTH must be set to 20 or 40. Valid settings are 16, 20, 32, or 40. See <a href="#">Interface Width Configuration, page 219</a> for more details. Width of valid data on Rdata and Sdata buses is the width of the internal datapath (16-bit or 20-bit). For the different possible bus widths, the previous and current valid Rdata and Sdata bits correspond to the following indices in ES_SDATA_MASK, ES_QUALIFIER, ES_QUAL_MASK, es_rdata, and es_sdata: <table style="margin-left: 20px;"> <tr><td>valid Rdata and Sdata width</td><td>previous data</td><td>current data</td></tr> <tr><td>16</td><td>[79:64]</td><td>[39:24]</td></tr> <tr><td>20</td><td>[79:60]</td><td>[39:20]</td></tr> </table>	valid Rdata and Sdata width	previous data	current data	16	[79:64]	[39:24]	20	[79:60]	[39:20]			
valid Rdata and Sdata width	previous data	current data												
16	[79:64]	[39:24]												
20	[79:60]	[39:20]												

**Table 4-20: RX Margin Analysis Attributes (Cont'd)**

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
RXOUT_DIV	Integer	PLL0 or PLL1 output clock divider D for the RX datapath as shown in <a href="#">Figure 2-9, page 36</a> . Valid settings are 1, 2, 4, or 8. This attribute sets the divider only if the RXRATE port is set to 3'b000.
USE_PCS_CLK_PHASE_SEL	1-bit Binary	If set to 1, Eye Scan 4T clock phase is determined by ES_CLK_PHASE_SEL. If set to 0, deserializer phase detector determines phase of Eye Scan 4T clock.
ES_CLK_PHASE_SEL	1-bit Binary	If USE_PCS_CLK_PHASE_SEL is asserted, setting to 1 selects one phase of Eye Scan 4T clock. Setting to 0 selects the other phase.

**Table 4-21: DRP Address Map for Eye Scan Read-Only (R) Registers**

<b>DRP Address Hex</b>	<b>DRP Bits</b>	<b>R/W</b>	<b>Name</b>	<b>Attribute Bit</b>
151	15:0	R	es_error_count	15:0
152	15:0	R	es_sample_count	15:0
153	3:0	R	es_control_status	3:0
154	15:0	R	es_rdata	79:64
155	15:0	R	es_rdata	63:48
156	15:0	R	es_rdata	47:32
157	15:0	R	es_rdata	31:16
158	15:0	R	es_rdata	15:0
159	15:0	R	es_sdata	79:64
15A	15:0	R	es_sdata	63:48
15B	15:0	R	es_sdata	47:32
15C	15:0	R	es_sdata	31:16
15D	15:0	R	es_sdata	15:0

## RX Polarity Control

### Functional Description

If RXP and RXN differential traces are accidentally swapped on the PCB, the differential data received by the GTP transceiver RX are reversed. The GTP transceiver RX allows inversion to be done on parallel bytes in the PCS after the SIPO to offset reversed polarity on differential pair. Polarity control function uses the RXPOLARITY input, which is driven High from the fabric user interface to invert the polarity.

### Ports and Attributes

[Table 4-22](#) defines the ports required by the RX polarity control function.

Table 4-22: RX Polarity Control Ports

Port	Dir	Clock Domain	Description
RXPOLARITY	In	RXUSRCLK2	The RXPOLARITY port can invert the polarity of incoming data: 0: Not inverted. RXP is positive and RXN is negative. 1: Inverted. RXP is negative and RXN is positive.

## Using RX Polarity Control

RXPOLARITY can be tied High if the polarity of RXP and RXN needs to be reversed.

## RX Pattern Checker

### Functional Description

The GTP transceiver receiver includes a built-in PRBS checker (see Figure 4-24). This checker can be set to check for one of four industry-standard PRBS patterns. The checker is self-synchronizing and works on the incoming data before comma alignment or decoding. This function can be used to test the signal integrity of the channel.

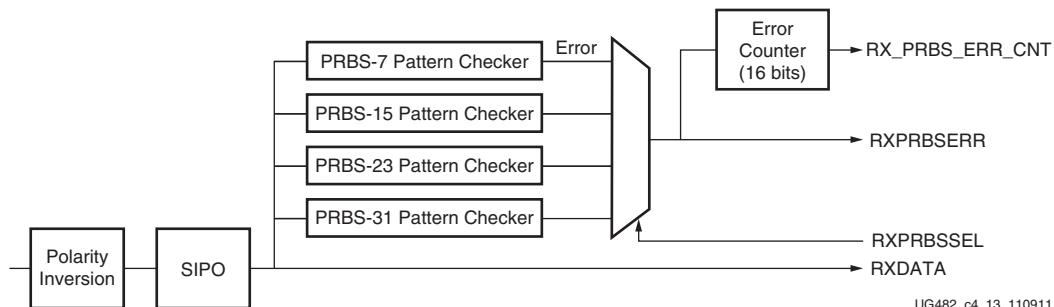


Figure 4-24: RX Pattern Checker Block

## Ports and Attributes

[Table 4-23](#) defines the pattern checker ports.

**Table 4-23: Pattern Checker Ports**

Port	Dir	Clock Domain	Description
RXPRBSCNTRESET	In	RXUSRCLK2	Resets the PRBS error counter.
RXPRBSSEL[2:0]	In	RXUSRCLK2	Receiver PRBS checker test pattern control. Only these settings are valid: 000: Standard operation mode. (PRBS check is off) 001: PRBS-7 010: PRBS-15 011: PRBS- 23 100: PRBS-31  No checking is done for non-PRBS patterns. Single bit errors cause bursts of PRBS errors because the PRBS checker uses data from the current cycle to generate next cycle's expected data.
RXPRBSERR	Out	RXUSRCLK2	This non-sticky status output indicates that PRBS errors have occurred.

[Table 4-24](#) defines the pattern checker attributes.

**Table 4-24: Pattern Checker Attributes**

Attribute	Type	Description
RX_PRBS_ERR_CNT	16-bit Binary	PRBS error counter. This counter can be reset by asserting RXPRBSCNTRESET. When an error(s) in incoming parallel data occurs, this counter increments by 1 and counts up to 0xFFFF. This error counter can only be accessed via the DRP. The address for this counter is 0x15E.
RXPRBS_ERR_LOOPBACK	1-bit Binary	When this attribute is set to 1, the RXPRBSERR bit is internally looped back to TXPRBSFORCEERR of the same GTP transceiver. This allows synchronous and asynchronous jitter tolerance testing without worrying about data clock domain crossing.  When this attribute is set to 0, TXPRBSFORCEERR is forced onto the TX PRBS.

## Use Models

To use the built-in PRBS checker, RXPRBSSEL must be set to match the PRBS pattern being sent to the receiver. The RXPRBSSEL entry in [Table 4-23](#) shows the available settings. When the PRBS checker is running, it attempts to find the selected PRBS pattern in the

incoming data. If the incoming data is inverted by the transmitter or reversed RXP/RXN, the received data should also be inverted by controlling RXPOLARITY. Otherwise, the PRBS checker does not lock. When it finds the pattern, it can detect PRBS errors by comparing the incoming pattern with the expected pattern. The expected pattern is generated from the previous incoming data. The checker counts the number of word (20 bits per word) errors and increments the word error counter by 1 when an error(s) is found in the incoming parallel data. Thus the word error counter might not match the actual number of bit errors if the incoming parallel data contains two or more bit errors. The error counter stops counting when reaching 0xFFFF.

When the error occurs, RXPRBSERR is asserted. When no error is found in the following incoming data, RXPRBSERR is cleared. Asserting RXPRBSCNTRESET clears the error counter. GTRXRESET and RXPCSRESET also reset the count.

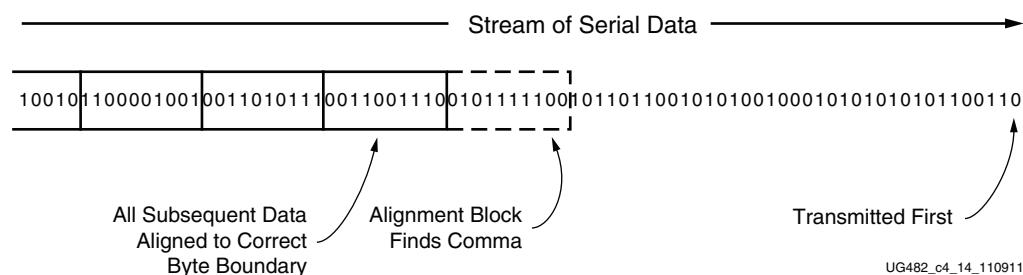
Refer to [TX Pattern Generator, page 105](#) for more information about use models.

## RX Byte and Word Alignment

### Functional Description

Serial data must be aligned to symbol boundaries before it can be used as parallel data. To make alignment possible, transmitters send a recognizable sequence, usually called a comma. The receiver searches for the comma in the incoming data. When it finds a comma, it moves the comma to a byte boundary so the received parallel words match the transmitted parallel words.

[Figure 4-25](#) shows the alignment to a 10-bit comma. The RX receiving unaligned bits are on the right side. The serial data with the comma is highlighted in the middle. Byte aligned RX parallel data is on the left.



*Figure 4-25: Conceptual View of Comma Alignment (Aligning to a 10-Bit Comma)*

Figure 4-26 shows TX parallel data on the left side, and RX receiving recognizable parallel data after comma alignment on the right side.

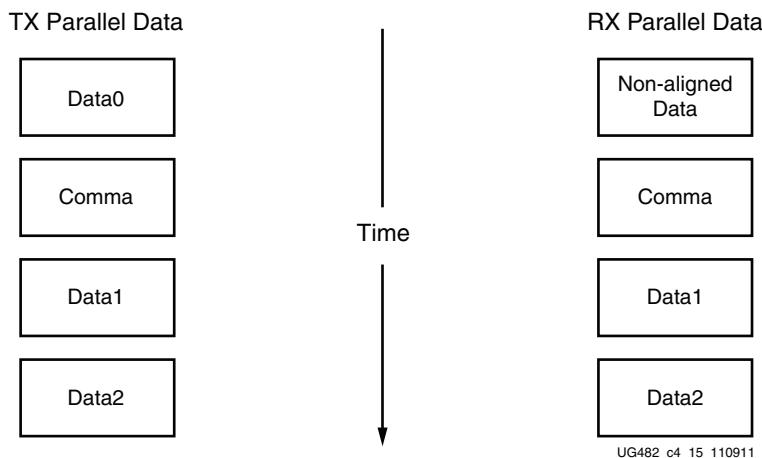


Figure 4-26: Parallel Data View of Comma Alignment

### Enabling Comma Alignment

To enable the comma alignment block, the RXCOMMADETEN port is driven High. RXCOMMADETEN is driven Low to bypass the block completely for minimum latency.

### Configuring Comma Patterns

To set the comma pattern that the block searches for in the incoming data stream, the ALIGN\_MCOMMA\_VALUE, ALIGN\_PCOMMA\_VALUE, and ALIGN\_COMMA\_ENABLE attributes are used. The comma lengths depend on RX\_DATA\_WIDTH (see Table 4-47, page 221). Figure 4-27 shows how the ALIGN\_COMMA\_ENABLE masks each of the comma values to allow partial pattern matching.

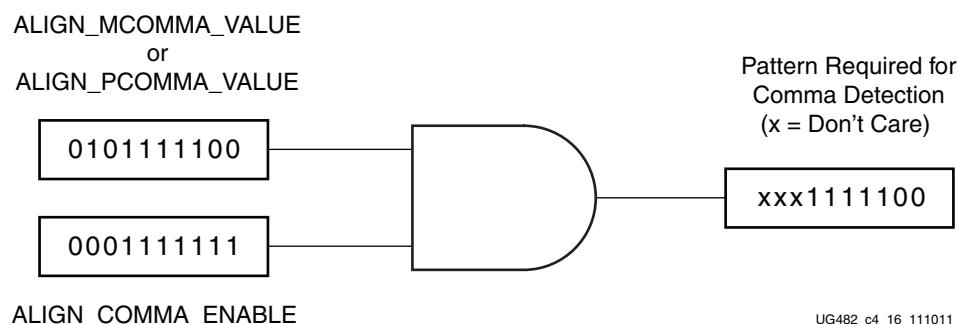
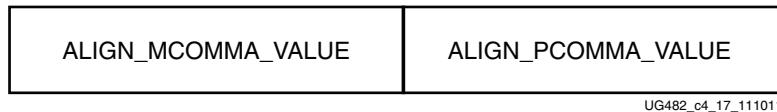


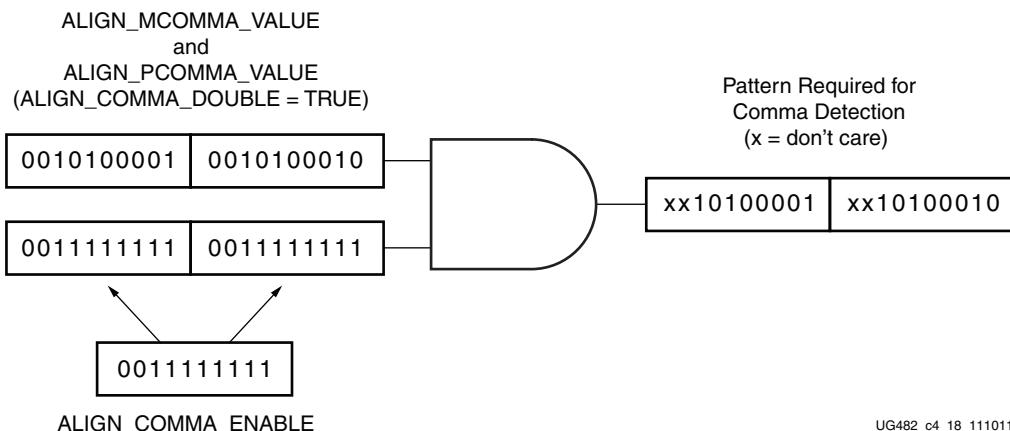
Figure 4-27: Comma Pattern Masking

[Figure 4-28](#) shows how the commas are combined when ALIGN\_COMMA\_DOUBLE is TRUE.



[Figure 4-28: Extended Comma Pattern Definition](#)

[Figure 4-29](#) shows how a comma is combined with ALIGN\_COMMA\_ENABLE to make a wild-carded comma for a 20-bit internal comma. If ALIGN\_COMMA\_DOUBLE is TRUE, the MCOMMA and PCOMMA patterns are combined so that the block searches for two commas in a row. The number of bits in the comma depends on RX\_DATA\_WIDTH. Either a 16-bit or a 20-bit comma alignment mode is possible. A double comma is only detected when the received data has a PCOMMA defined by ALIGN\_PCOMMA\_VALUE followed by an MCOMMA defined by ALIGN\_MCOMMA\_VALUE with no extra bits in between.



[Figure 4-29: Extended Comma Pattern Masking](#)

### Activating Comma Alignment

Commas are aligned to the closest boundary providing they are found while comma alignment is active. RXMCOMMAALIGNEN is driven High to align on the MCOMMA pattern. RXPCOMMAALIGNEN is driven High to activate alignment on the PCOMMA pattern. Both enable ports are driven to align to either pattern. When ALIGN\_COMMA\_DOUBLE is TRUE, both enable ports must always be driven to the same value.

### Alignment Status Signals

While MCOMMA or PCOMMA alignment is active, any matching comma pattern causes the block to realign to the closest boundary. After successful alignment, the block holds RXBYTEISALIGNED High. At this time, RXMCOMMAALIGNEN and RXPCOMMAALIGNEN can be driven Low to turn off alignment and keep the current alignment position. RXPCOMMAALIGNEN must be TRUE for PCOMMAS to cause RXBYTEISALIGNED to go High. Similarly, RXMCOMMAALIGNEN must be TRUE for MCOMMAS to cause RXBYTEISALIGNED to go High. Commas can arrive while RXBYTEISALIGNED is High. If the commas arrive aligned to boundaries, there is no change. If the commas arrive out of position, the block deasserts RXBYTEISALIGNED

until the commas are aligned again. If alignment is still activated for the comma that arrives, the block automatically aligns the new comma to the closest boundary and drives RXBYTEREALIGN High for one RXUSRCLK2 cycle.

In applications that operate at a line rate greater than 5 Gb/s and have excessive noise in the system, the byte align block might falsely align to a wrong byte boundary and falsely assert the RXBYTEISALIGNED signal when no valid data is present. In such applications, a system level check should be in place for checking the validity of the RXBYTEISALIGNED indicator and data.

In systems that use the OOB (RX Out-of-Band signaling) block e.g., PCIe, SATA, after locking to a valid byte boundary and asserting the RXBYTEISALIGNED signal, the byte align block might occasionally de-assert the RXBYTEISALIGNED signal even when there is no change in the byte boundary. In such applications, RXBYTEISALIGNED should not be used as a valid indicator of the change in byte boundary after the first assertion.

### Alignment Boundaries

The allowed boundaries for alignment are defined by ALIGN\_COMM WORD. The spacing of the possible boundaries is determined by RX\_DATA\_WIDTH, and the number of boundary positions is determined by the number of bytes in the RXDATA interface (refer to [Table 4-43, page 219](#) for RX\_DATA\_WIDTH settings). [Figure 4-30](#) shows the boundaries that can be selected.

RX_DATA_WIDTH	ALIGN_COMM WORD	Possible RX Alignments (Grey = Comma Can Appear on Byte)
16/20 (2-byte)	1	Byte1 Byte0
16/20 (2-byte)	2	Byte1 Byte0
32/40 (4-byte)	1	Byte3 Byte2 Byte1 Byte0
32/40 (4-byte)	2	Byte3 Byte2 Byte1 Byte0

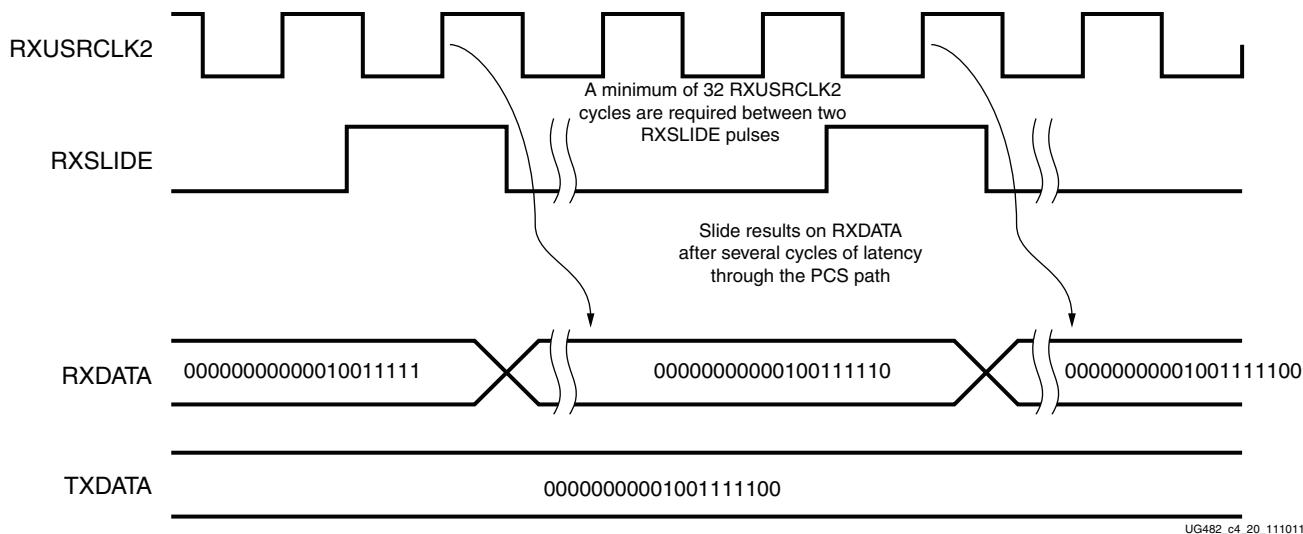
UG482\_c4\_19\_112811

*Figure 4-30: Comma Alignment Boundaries*

## Manual Alignment

RXSLIDE can be used to override the automatic comma alignment and to shift the parallel data. RXSLIDE is driven High for one RXUSRCLK2 cycle to shift the parallel data by one bit. RXSLIDE must be Low for at least 32 RXUSRCLK2 cycles before it can be used again.

[Figure 4-31](#) shows the waveforms for manual alignment using RXSLIDE in RXSLIDE\_MODE = PCS, before and after the data shift. When RXSLIDE\_MODE = PCS is used, the number of bit shift positions when consecutive RXSLIDE pulses are issued is also determined by the comma alignment boundary set by ALIGN\_COMMA\_WORD and RX\_DATA\_WIDTH. For example, if the RX\_DATA\_WIDTH is 20 bits and ALIGN\_COMMA\_WORD is 1, after the 9th slide operation, the slide position returns back to 0. For the same RX\_DATA\_WIDTH setting, for an ALIGN\_COMMA\_WORD setting of 2, the slide position returns to 0 after the 19th slide operation.

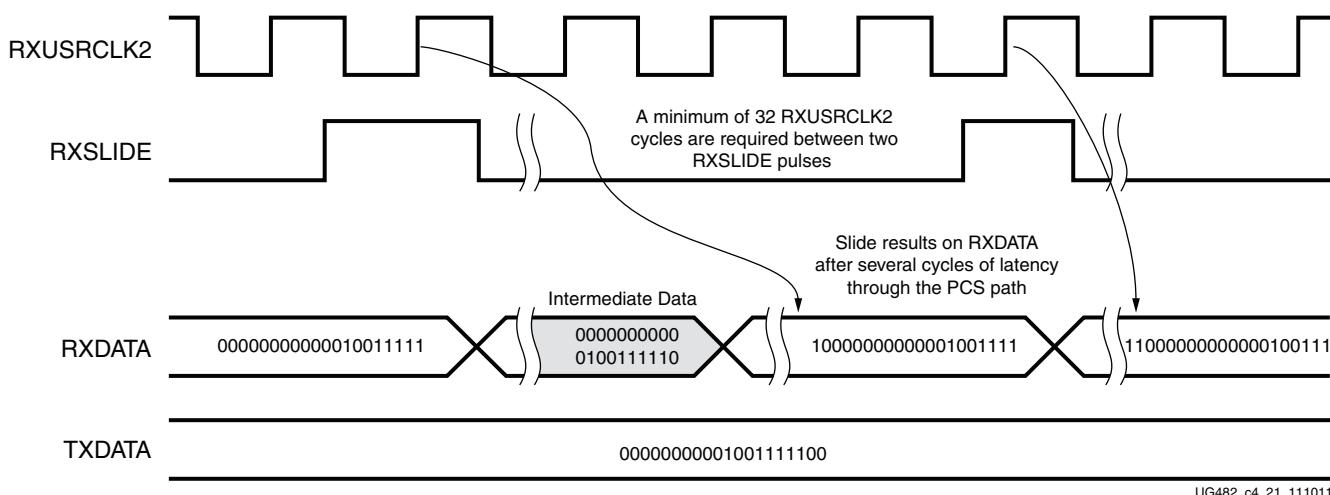


**Figure 4-31: Manual Data Alignment Using RXSLIDE for RX\_DATA\_WIDTH = 20 Bits and RXSLIDE\_MODE = PCS**

Note relevant to [Figure 4-31](#):

1. Latency between the slide and the slide result at RXDATA depends on the number of active RX PCS blocks in the datapath.

[Figure 4-32](#) shows the waveforms for manual alignment using RXSLIDE in RXSLIDE\_MODE = PMA before and after the data shift. In this mode, the data is shifted right by one bit for every RXSLIDE pulse issued, but there is some intermediate data with the bits shifted left before the final data appears on the bus. When RXSLIDE\_MODE = PMA is used, the RX recovered clock phase is shifted by 2 UI for every alternate RXSLIDE pulse.



UG482\_c4\_21\_111011

**Figure 4-32: Manual Data Alignment Using RXSLIDE for RX\_DATA\_WIDTH = 20 Bits and RXSLIDE\_MODE = PMA**

Note relevant to [Figure 4-32](#):

1. Latency between the slide and the slide result at RXDATA depends on the number of active RX PCS blocks in the datapath.

## Ports and Attributes

Table 4-25 defines the RX byte and word alignment ports.

Table 4-25: RX Byte and Word Alignment Ports

Port Name	Dir	Clock Domain	Description
RXBYTEISALIGNED	Out	RXUSRCLK2	<p>This signal from the comma detection and realignment circuit is High to indicate that the parallel data stream is properly aligned on byte boundaries according to comma detection.</p> <ul style="list-style-type: none"> <li>0: Parallel data stream not aligned to byte boundaries</li> <li>1: Parallel data stream aligned to byte boundaries</li> </ul> <p>There are several cycles after RXBYTEISALIGNED is asserted before aligned data is available at the FPGA RX interface.</p> <p>RXBYTEISALIGNED responds to plus comma alignment when RXPCOMMAALIGNEN is TRUE. RXBYTEISALIGNED responds to minus comma alignment when RXMCOMMAALIGNEN is TRUE.</p> <p>Some conditions when this signal could deviate from the expected behavior are discussed in <a href="#">Alignment Status Signals, page 166</a>.</p>
RXBYTEREALIGN	Out	RXUSRCLK2	<p>This signal from the comma detection and realignment circuit indicates that the byte alignment within the serial data stream has changed due to comma detection.</p> <ul style="list-style-type: none"> <li>0: Byte alignment has not changed</li> <li>1: Byte alignment has changed</li> </ul> <p>Data can be lost or repeated when alignment occurs, which can cause data errors (and disparity errors when the 8B/10B decoder is used).</p>
RXCOMMADET	Out	RXUSRCLK2	<p>This signal is asserted when the comma alignment block detects a comma. The assertion occurs several cycles before the comma is available at the FPGA RX interface.</p> <ul style="list-style-type: none"> <li>0: Comma not detected</li> <li>1: Comma detected</li> </ul>
RXCOMMADETEN	In	RXUSRCLK2	<p>RXCOMMADETEN activates the comma detection and alignment circuit.</p> <ul style="list-style-type: none"> <li>0: Bypass the circuit</li> <li>1: Use the comma detection and alignment circuit</li> </ul> <p>Bypassing the comma and alignment circuit reduces RX datapath latency.</p>
RXPCOMMAALIGNEN	In	RXUSRCLK2	<p>Aligns the byte boundary when comma plus is detected.</p> <ul style="list-style-type: none"> <li>0: Disabled</li> <li>1: Enabled.</li> </ul>
RXMCOMMAALIGNEN	In	RXUSRCLK2	<p>Aligns the byte boundary when comma minus is detected.</p> <ul style="list-style-type: none"> <li>0: Disabled</li> <li>1: Enabled.</li> </ul>

Table 4-25: RX Byte and Word Alignment Ports (Cont'd)

Port Name	Dir	Clock Domain	Description
RXSLIDE	In	RXUSRCLK2	<p>RXSLIDE implements a comma alignment bump control. When RXSLIDE is asserted, the byte alignment is adjusted by one bit, which permits determination and control of byte alignment by the FPGA logic. Each assertion of RXSLIDE causes just one adjustment.</p> <p>RXSLIDE must be deasserted for more than 32 RXUSRCLK2 cycles before it can be reasserted to cause another adjustment.</p> <p>When asserted, RXSLIDE takes precedence over normal comma alignment.</p> <p>For proper operation, the user should set these values:</p> <ul style="list-style-type: none"> <li>RXPCommaAlignen = 0;</li> <li>RXMCommaAlignen = 0;</li> <li>RXCommaDeten = 1;</li> <li>SHOW_REALIGN_COMMA = FALSE</li> </ul>

Table 4-26 defines the RX byte and word alignment attributes.

Table 4-26: RX Byte and Word Alignment Attributes

Attribute	Type	Description
ALIGN_COMM WORD	Integer	<p>This attribute controls the alignment of detected commas within a multi-byte datapath.</p> <p>1: Align comma to either of the 2 bytes for a 2-byte interface, any of the 4 bytes for a 4-byte interface.</p> <p>The comma can be aligned to either the even bytes or the odd bytes of RXDATA output.</p> <p>2: Align comma to the even bytes only. The aligned comma is guaranteed to be aligned to even bytes RXDATA[9:0] for a 2-byte interface, RXDATA[9:0]/RXDATA[29:20] for a 4-byte interface.</p> <p>Refer to <a href="#">Figure 4-30, page 167</a> for comma alignment boundaries allowed for the different ALIGN_COMM WORD and RX_DATA_WIDTH settings.</p> <p>Protocols that send commas in even and odd positions must set ALIGN_COMM WORD to 1.</p>
ALIGN_COMM_ENABLE	10-bit Binary	<p>Sets which bits in MCOMMA/PCOMMA must be matched to incoming data and which bits can be of any value.</p> <p>This attribute is a 10-bit mask with a default value of 1111111111. Any bit in the mask that is reset to 0 effectively turns the corresponding bit in MCOMMA or PCOMMA to a don't care bit.</p>

Table 4-26: RX Byte and Word Alignment Attributes (Cont'd)

Attribute	Type	Description
ALIGN_COMMA_DOUBLE	String	<p>Specifies whether a comma match consists of either a comma plus or a comma minus alone, or if both are required in the sequence.</p> <p>FALSE: The plus comma (PCOMMA) and minus comma (MCOMMA) are handled separately. An individual match for either can lead to comma detection and alignment.</p> <p>TRUE: A comma match consists of a comma plus followed immediately by a comma minus. The match pattern is 20 or 16 bits (as determined by RX_DATA_WIDTH).</p> <p>When ALIGN_COMMA_DOUBLE is TRUE, ALIGN_PCOMMA_DET must be the same as ALIGN_MCOMMA_DET, and RXPCOMMAALIGNEN must be the same as RXMCOMMAALIGNEN.</p>
ALIGN_MCOMMA_VALUE	10-bit Binary	Defines comma minus to raise RXCOMMADET and align the parallel data. The reception order is right to left. (ALIGN_MCOMMA_VALUE [0] is received first.) The default value is 10'b101000011 (K28.5). This definition does not affect 8B/10B encoding or decoding.
ALIGN_MCOMMA_DET	String	<p>Controls the raising of RXCOMMADET on comma minus.</p> <p>FALSE: Do not raise RXCOMMADET when comma minus is detected.</p> <p>TRUE: Raise RXCOMMADET when comma minus is detected. (This setting does not affect comma alignment.)</p>
ALIGN_PCOMMA_VALUE	10-bit Binary	Defines comma plus to raise RXCOMMADET and align parallel data. The reception order is right to left. (ALIGN_PCOMMA_VALUE [0] is received first.) The default value is 10'b010111100 (K28.5). This definition does not affect 8B/10B encoding or decoding.
ALIGN_PCOMMA_DET	String	<p>Controls the raising of RXCOMMADET on comma plus.</p> <p>FALSE: Do not raise RXCOMMADET when comma plus is detected.</p> <p>TRUE: Raise RXCOMMADET when comma plus is detected. (This setting does not affect comma alignment.)</p>
SHOW_REALIGN_COMMA	String	<p>Defines if a comma that caused realignment is brought out to the FPGA RX.</p> <p>FALSE: Do not bring the comma that causes realignment to the FPGA RX. This setting reduces RX datapath latency</p> <p>TRUE: Bring the realignment comma to the FPGA RX.</p> <p>SHOW_REALIGN_COMMA = TRUE should not be used when ALIGN_COMMA_DOUBLE = TRUE or when manual alignment is used.</p>

Table 4-26: RX Byte and Word Alignment Attributes (*Cont'd*)

Attribute	Type	Description
RXSLIDE_MODE	String	Defines the RXSLIDE mode. OFF: Default setting. The RXSLIDE feature is not used. PCS: PCS is used to perform the bit-slipping function. RXSLIDE is driven High for one RXUSRCLK2 cycle to shift the parallel data (RXDATA) to the left by one bit within the comma alignment boundary determined by the ALIGN_COMMA_WORD and RX_DATA_WIDTH settings. In this mode, even if RXOUTCLK is sourcing from the RX PMA, the clock phase remains the same. This option requires SHOW_REALIGN_COMMA to be FALSE. PMA: PMA is used to perform the bit-slipping function. RXSLIDE is driven High for one RXUSRCLK2 cycle to shift the parallel data (RXDATA) to the right by one bit. If RXOUTCLK is sourcing from the RX PMA, its phase might be changed. This mode provides minimum variation of latency compared to PCS mode. This option requires SHOW_REALIGN_COMMA to be FALSE. AUTO: This is an automated PMA mode without using the FPGA logic to monitor the RXDATA and issue RXSLIDE pulses. In this mode, RXSLIDE is ignored. In PCIe applications, this setting is used for FTS lane deskew. This option requires SHOW_ALIGN_COMMA to be FALSE.
RXSLIDE_AUTO_WAIT	Integer	Defines how long the PCS (in terms of RXUSRCLK clock cycle) waits for the PMA to auto slide before checking the alignment again. Valid settings are from 0 to 15. The default value is 7. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RX_SIG_VALID_DLY	Integer	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
COMMA_ALIGN_LATENCY	6-bit Binary	Current alignment used by the byte align block to align the incoming data based on the comma location locked. This register is only accessible via the DRP: Bits[6:0] of DRP address 0x150

## RX 8B/10B Decoder

### Functional Description

If RX received data is 8B/10B encoded, it must be decoded. The GTP transceiver has a built-in 8B/10B encoder in the GTP transceiver TX and an 8B/10B decoder in the GTP transceiver RX, which includes two one-byte 8B/10B decoder modules on the datapath to decode data without consuming FPGA resources. The RX 8B/10B decoder has these features:

1. Supports 2-byte and 4-byte datapath operation
2. Provides daisy-chained hookup of running disparity for proper disparity
3. Generates K characters and status outputs
4. Can be bypassed if incoming data is not 8B/10B encoded
5. Pipes out 10-bit literal encoded values when encountering a not-in-table error

## 8B/10B Bit and Byte Ordering

The order of the bits into the 8B/10B decoder is the opposite of the order shown in [Appendix C, 8B/10B Valid Characters](#). 8B/10B decoding requires bit a0 to be received first, but the GTP transceiver always receives the right-most bit first. Consequently, the 8B/10B decoder automatically reverses the bit order of received data before decoding it. Decoded data is available on RXDATA ports. [Figure 4-33](#) shows data received by the GTP transceiver RX when RX\_DATA\_WIDTH = 20 or 40. Data is reconstructed into bytes and sent to the RXDATA interface after the 8B/10B decoder. The number of bits used by RXDATA and corresponding byte orders are determined by RX\_DATA\_WIDTH.

- Only use RXDATA[15:0] if RX\_DATA\_WIDTH = 20
- Use full RXDATA[31:0] if RX\_DATA\_WIDTH = 40

When the 8B/10B decoder is bypassed but RX\_DATA\_WIDTH is set to multiple of 10, 10-bit characters are passed to the RX data interface with this format:

- The corresponding RXDISPERR represents the 9th bit
- The corresponding RXCHARISK represents the 8th bit
- The corresponding RXDATA byte represents the [7:0] bits

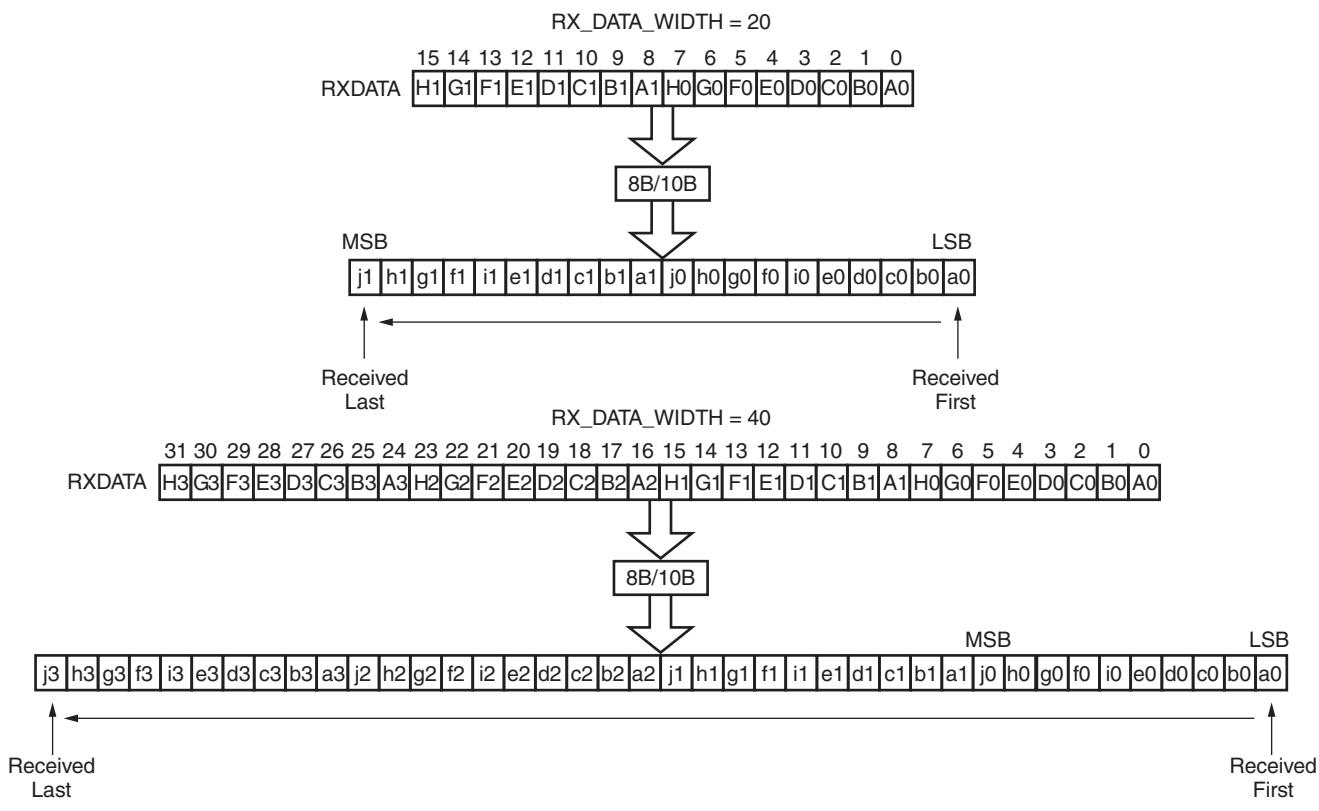


Figure 4-33: 8B/10B Decoder Bit and Byte Order

## RX Running Disparity

Disparity check is performed and the decoder drives the corresponding RXDISPERR High when the data byte on RXDATA arrives with the wrong disparity. In addition to disparity errors, the 8B/10B decoder detects 20-bit out-of-table error codes. The decoder drives the

RXNOTINTABLE port High when decoder is enabled but a received 10-bit character cannot be mapped into a valid 8B/10B character listed in [Appendix C, 8B/10B Valid Characters](#). The non-decoded 10-bit character is piped out of the decoder through the RX data interface with this format:

- The corresponding RXDISPERR represents the 9th bit
- The corresponding RXCHARISK represents the 8th bit
- The corresponding RXDATA byte represents the [7:0] bits

[Figure 4-34](#) shows a waveform at the RX data interface when the decoder receives good data (A), data with disparity error (B), an out-of-table character (C), and an out-of-table character with disparity error (D).

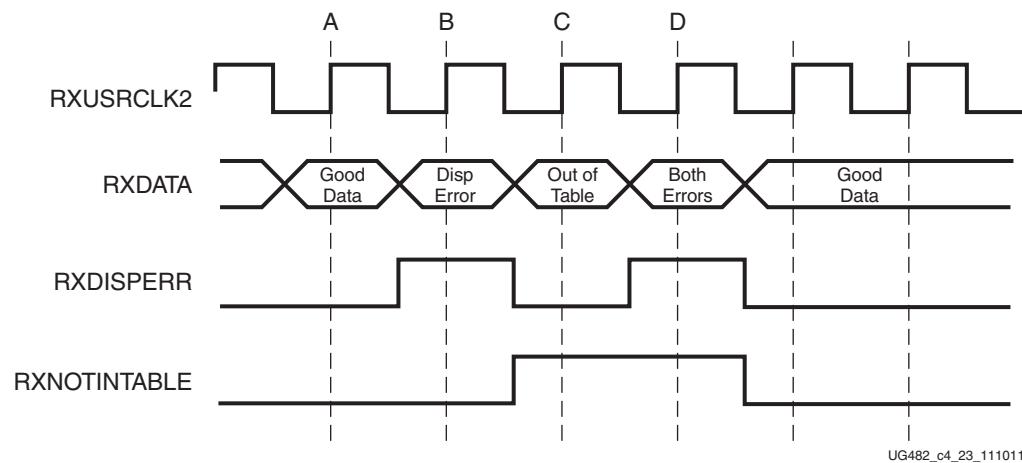


Figure 4-34: RX Data with 8B/10B Errors

## Special Characters

8B/10B decoding includes special characters (K characters) that are often used for control functions. When RXDATA is a K character, the decoder drives RXCHARISK High.

If DEC\_PCOMMA\_DETECT is set to TRUE, the decoder drives the corresponding RXCHARISCOMMA High whenever RXDATA is a positive 8B/10B comma. If DEC\_MCOMMA\_DETECT is TRUE, the decoder drives the corresponding RXCHARISCOMMA bit High whenever RXDATA is a negative 8B/10B comma.

## Ports and Attributes

Table 4-27 defines the ports required by RX 8B/10B decoder.

Table 4-27: RX 8B/10B Decoder Ports

Port	Dir	Clock Domain	Description
RX8B10BEN	In	RXUSRCLK2	RX8B10BEN selects the use of the 8B/10B decoder in the RX datapath, just after the comma detection/realignment block. If this input is Low, the literal 10-bit data comes out as {RXDISPERR, RXCHARISK, RXDATA<8 bits>}. 1: 8B/10B decoder enabled 0: 8B/10B decoder bypassed (reduces latency)
RXCHARISCOMMA[3:0]	Out	RXUSRCLK2	Active High indicates the corresponding byte shown on RXDATA is a comma character. RXCHARISCOMMA[3] corresponds to RXDATA[31:24] RXCHARISCOMMA[2] corresponds to RXDATA[23:16] RXCHARISCOMMA[1] corresponds to RXDATA[15:8] RXCHARISCOMMA[0] corresponds to RXDATA[7:0]
RXCHARISK[3:0]	Out	RXUSRCLK2	Active High indicates the corresponding byte shown on RXDATA is a K character when 8B/10B decoding is enabled. RXCHARISK[3] corresponds to RXDATA[31:24] RXCHARISK[2] corresponds to RXDATA[23:16] RXCHARISK[1] corresponds to RXDATA[15:8] RXCHARISK[0] corresponds to RXDATA[7:0] This is bit 8 of non-decoded data if the 8B/10B decoder is bypassed or the corresponding bit of RXNOTINTABLE is High. Refer to <a href="#">FPGA RX Interface, page 218</a> .
RXDISPERR[3:0]	Out	RXUSRCLK2	Active High indicates the corresponding byte shown on RXDATA has a disparity error. RXDISPERR[3] corresponds to RXDATA[31:24] RXDISPERR[2] corresponds to RXDATA[23:16] RXDISPERR[1] corresponds to RXDATA[15:8] RXDISPERR[0] corresponds to RXDATA[7:0] This is bit 9 of non-decoded data if the 8B/10B decoder is bypassed or the corresponding bit of RXNOTINTABLE is High. Refer to <a href="#">FPGA RX Interface, page 218</a> .
RXNOTINTABLE[3:0]	Out	RXUSRCLK2	Active High indicates the corresponding byte shown on RXDATA was not a valid character in the 8B/10B table. RXNOTINTABLE[3] corresponds to RXDATA[31:24] RXNOTINTABLE[2] corresponds to RXDATA[23:16] RXNOTINTABLE[1] corresponds to RXDATA[15:8] RXNOTINTABLE[0] corresponds to RXDATA[7:0]

Table 4-28: RX 8B/10B Decoder Attributes

Attribute	Type	Description
RX_DISPERR_SEQ_MATCH	StringString	Specifies whether the disparity error status of a decoded byte must match the indicator in the channel bonding and clock correction sequence. When TRUE, indicates the disparity error status must be matched. When FALSE, ignores the disparity error status.
DEC_MCOMMA_DETECT	String	When set to TRUE, drives the per byte flag RXCHARISCOMMA High when an MCOMMA is detected. When set to FALSE, RXCHARISCOMMA is Low when a negative comma detected.
DEC_PCOMMA_DETECT	String	When set to TRUE, drives the per byte flag RXCHARISCOMMA High when a PCOMMA is detected. When set to FALSE, RXCHARISCOMMA is Low when a positive comma detected.
DEC_VALID_COMMA_ONLY	String	When set to TRUE, drives the per byte flag RXCHARISCOMMA High when only IEEE 802.3 valid commas K28.1, K28.5, and K28.7 are detected. When set to FALSE, RXCHARISCOMMA is for positive or negative 8B/10B commas, depending how the user sets DEC_PCOMMA_DETECT and DEC_MCOMMA_DETECT.
RX_DATA_WIDTH	3-bit Binary	The PCS data width is set at the Fabric user interface with values of 16 or 32 (if 8B/10B decoding is not used) or 20 or 40 (if 8B/10B decoding is used).

## Enabling and Disabling 8B/10B Decoding

To enable the 8B/10B decoder, RX8B10BEN must be driven High. RX\_DATA\_WIDTH must be set to a multiple of 8 (8, 16, 32) when the 8B/10B decoder enabled.

To disable the 8B/10B decoder on the GTP transceiver receiver path, RX8B10BEN must be driven Low. When the encoder is disabled, RX\_DATA\_WIDTH can be set to a multiple of 10 (10, 20, 40). The operation of the RXDATA port with 8B/10B decoding bypassed is described in [FPGA RX Interface, page 218](#).

## RX Buffer Bypass

### Functional Description

Bypassing the RX elastic buffer is an advanced feature of the 7 series GTP transceiver. The RX phase alignment circuit is used to adjust the phase difference between the PMA parallel clock domain (XCLK) and the RXUSRCLK domain when the RX elastic buffer is bypassed. It also performs the RX delay alignment by adjusting the RXUSRCLK to compensate for the temperature and voltage variations. The combined RX phase and delay alignments can be automatically performed by the GTP transceiver or manually controlled by the user. [Figure 4-43](#) shows the XCLK and RXUSRCLK domains, and [Table 4-32](#) shows trade-offs between buffering and phase alignment.

The RX elastic buffer can be bypassed to reduce latency when the RX recovered clock is used to source RXUSRCLK and RXUSRCLK2. When the RX elastic buffer is bypassed, latency through the RX datapath is low and deterministic, but clock correction and channel bonding are not available.

When RX buffer bypass is used, RXSLIDE\_MODE cannot be set to AUTO or PMA.

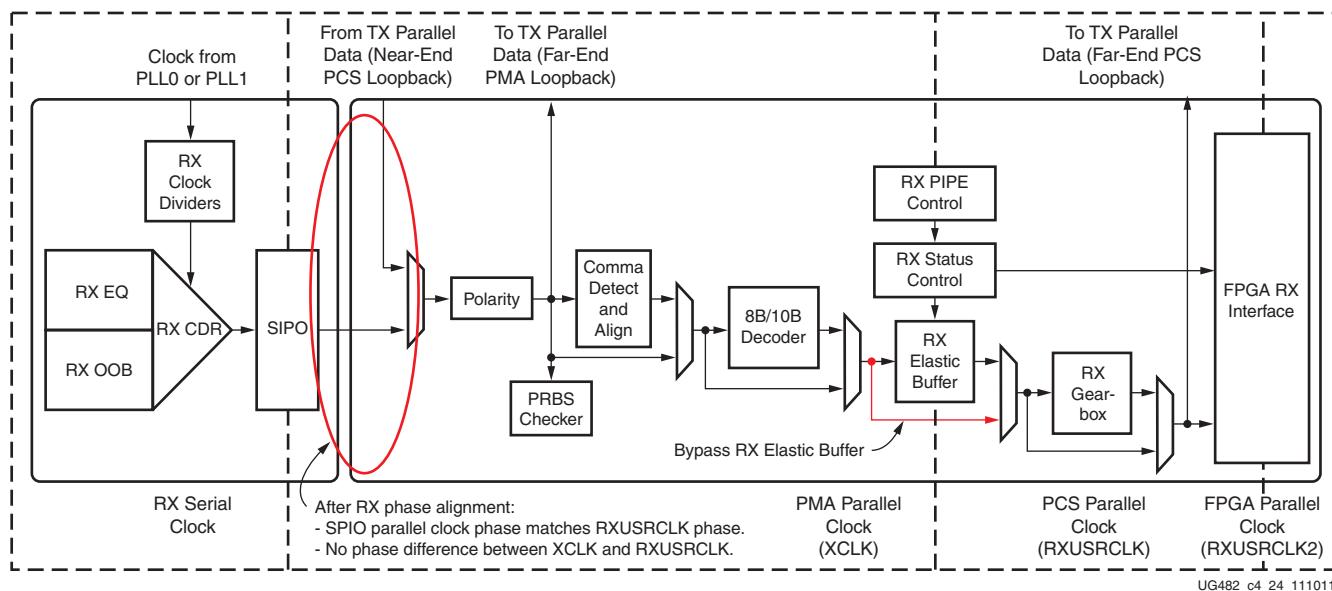


Figure 4-35: Using RX Phase Alignment

## Ports and Attributes

Table 4-29 defines the RX buffer bypass ports.

Table 4-29: RX Buffer Bypass Ports

Port	Dir	Clock Domain	Description
RXPHDLYRESET	In	Async	RX phase alignment hard reset to force RXUSRCLK to the center of the delay alignment tap. The delay alignment tap has a full range of $\pm 4$ ns and a half range of $\pm 2$ ns. This hard reset can be used to initiate the GTP transceiver to perform the RX phase and delay alignment automatically when all other RX buffer bypass input ports are set Low. It is recommended to use RXDLYSRESET only for phase and delay alignment.
RXPHALIGN	In	Async	Sets the RX phase alignment. Tied Low when using the auto alignment mode.
RXPHALIGNEN	In	Async	RX phase alignment enable. Tied Low when using the auto alignment mode.

Table 4-29: RX Buffer Bypass Ports (Cont'd)

Port	Dir	Clock Domain	Description
RXPHDLYPD	In	Async	RX phase and delay alignment circuit power down. Tied High when a) RX buffer bypass is not in use; b) RXPD is asserted or c) RXOUTCLKSEL is set to 3'b010 but the recovered clock is not available. Tied Low during RX buffer bypass mode normal operation. 0: Power-up the RX phase and delay alignment circuit. 1: Power-down the RX phase and delay alignment circuit.
RXPHOVRDEN	In	Async	RX phase alignment counter override enable. Tied Low when not in use. 0: Normal operation. 1: Enables the RX phase alignment counter override with the RXPH_CFG[10:6] value.
RXDLYSRESET	In	Async	RX delay alignment soft reset to gradually shift RXUSRCLK to the center of the delay alignment tap. The delay alignment tap has a full range of ±4 ns and a half range of ±2 ns. This soft reset can be used to initiate the GTP transceiver to perform the RX phase and delay alignment automatically when all other RX bypass buffer input ports are Low.
RXDLYBYPASS	In	Async	RX delay alignment bypass. 0: Uses the RX delay alignment circuit. 1: Bypasses the RX delay alignment circuit.
RXDLYEN	In	Async	RX delay alignment enable. Tied Low when not in use.
RXDLYOVRDEN	In	Async	RX delay alignment counter override enable. Tied Low when not in use. 0: Normal operation. 1: Enables the RX delay alignment counter override with the RXDLY_CFG[14:6] value.
RXDDIEN	In	Async	RX data delay insertion enable in the deserializer. Set High in RX buffer bypass mode.

Table 4-29: RX Buffer Bypass Ports (Cont'd)

Port	Dir	Clock Domain	Description
RXPHALIGNDONE	Out	Async	RX phase alignment done. When the auto RX phase and delay alignment are used, the second rising edge of RXPHALIGNDONE detected after RXDLYSRESETDONE assertion indicates RX phase and delay alignment are done.
RXPHMONITOR	Out	Async	RX phase alignment monitor.
RXPHSLIPMONITOR	Out	Async	RX phase alignment slip monitor.
RXDLYSRESETDONE	Out	Async	RX delay alignment soft reset done.
RXSYNCMODE	In	Async	0: RX Buffer Bypass Slave lane 1: RX Buffer Bypass Master lane This input is not used in multi-lane manual mode.
RXSYNCALLIN	In	Async	Single-lane auto mode: Connect this input to its own RXPHALIGNDONE. Multi-lane auto mode: Connect this input to the ANDed signal of RXPHALIGNDONE of the master and all slave lanes. Multi-lane manual mode: This input is not used in multi-lane manual mode.
RXSYNCIN	In	Async	Only valid in multi-lane auto mode applications. Connect this input to RXSYNCOUT from RX buffer bypass master lane.
RXSYNCOUT	Out	Async	Only valid for RX buffer bypass master lane in multi-lane auto mode applications. Connect this signal to the RXSYNCIN of each lane within the multi-lane application.
RXSYNCDONE	Out	Async	Indicates RX Buffer Bypass alignment procedure completion. Only valid for RX buffer bypass master lane in auto mode operation.

Table 4-30 defines the RX buffer attributes.

Table 4-30: RX Buffer Bypass Attributes

Attribute	Type	Description
RXBUF_EN	String	Use or bypass the RX elastic buffer. TRUE: Uses the RX elastic buffer (default). FALSE: Bypasses the RX elastic buffer (advanced feature).

Table 4-30: RX Buffer Bypass Attributes (Cont'd)

Attribute	Type	Description
RX_XCLK_SEL	String	Selects the clock source used to drive the RX parallel clock domain (XCLK). RXREC: Selects the RX recovered clock as source of XCLK. Used when using the RX elastic buffer. RXUSR: Selects RXUSRCLK as the source of XCLK. Used when bypassing the RX elastic buffer.
RXPH_CFG	24-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXPH_MONITOR_SEL	5-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXPHDLY_CFG	24-bit Binary	RX phase and delay alignment configuration. RXPHDLY_CFG[19] = 1 is used to set the RX delay alignment tap to the full range of $\pm 4$ ns. RXPHDLY_CFG[19] = 0 is used to set the RX delay alignment tap to the half range of $\pm 2$ ns. Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXDLY_CFG	16-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXDLY_LCFG	9-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXDLY_TAP_CFG	16-bit Binary	Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RX_DDI_SEL	6-bit Binary	RX data delay insertion select. Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXSYNC_MULTILANE	1-bit Binary	Indicates whether the lane is used as part of a multi-lane interface. Only valid on RX buffer bypass master lane in auto mode. 0: This lane is used in single-lane mode. 1: This lane is used in multi-lane mode.
RXSYNC_SKIP_DA	1-bit Binary	Control to skip delay alignment procedure. Only valid on buffer bypass master lane in auto mode. 0: RX delay alignment procedure occurs. 1: RX delay alignment procedure is skipped.

Table 4-30: RX Buffer Bypass Attributes (Cont'd)

Attribute	Type	Description
RXSYNC_OVRD	1-bit Binary	Manual mode override. 0: RX Buffer bypass auto mode is enabled. 1: RX Buffer bypass manual mode is used. RX Buffer bypass control is implemented in fabric logic.
TST_RSV[0]	1-bit Binary	0: Normal. 1: Override data delay insertion (DDI) delay setting with RX_DDI_SEL attribute.

## RX Buffer Bypass Use Modes

RX phase alignment can be performed on one channel (single lane) or a group of channels sharing a single RXOUTCLK (multi-lane). For GTP transceivers, RX buffer bypass supports single-lane auto mode, and multi-lane applications in manual and auto mode (Table 4-31).

Table 4-31: RX Buffer Bypass Use Modes

Rx Buffer Bypass	GTP Transceiver
Single-Lane	Auto
Multi-Lane	Manual or Auto

## Using RX Buffer Bypass in Single-Lane Auto Mode

These GTP transceiver settings should be used to bypass the RX buffer:

- RXBUF\_EN = FALSE.
- RX\_XCLK\_SEL = RXUSR.
- RXOUTCLKSEL = 010b to select the RX recovered clock as the source of RXOUTCLK.
- RXDDIEN = 1.

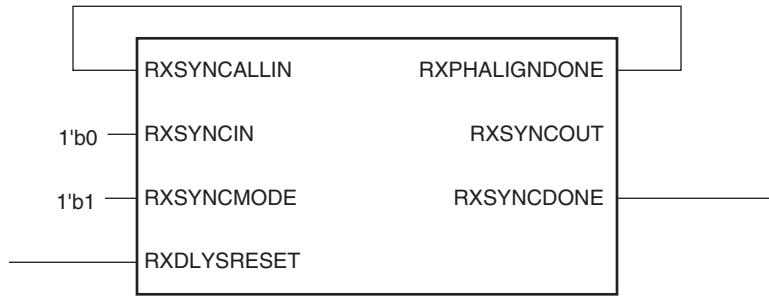
With the RX recovered clock selected, RXOUTCLK is to be used as the source of RXUSRCLK. The user must ensure that RXOUTCLK and the selected RX recovered clock are running and operating at the desired frequency. When the RX elastic buffer is bypassed, the RX phase alignment procedure must be performed after these conditions:

- Resetting or powering up the GTP receiver.
- Resetting or powering up the PLL.
- Changing the RX recovered clock source or frequency.
- Changing the GTP RX line rate.

To set up RX buffer bypass in single-lane auto mode, these attributes should be set:

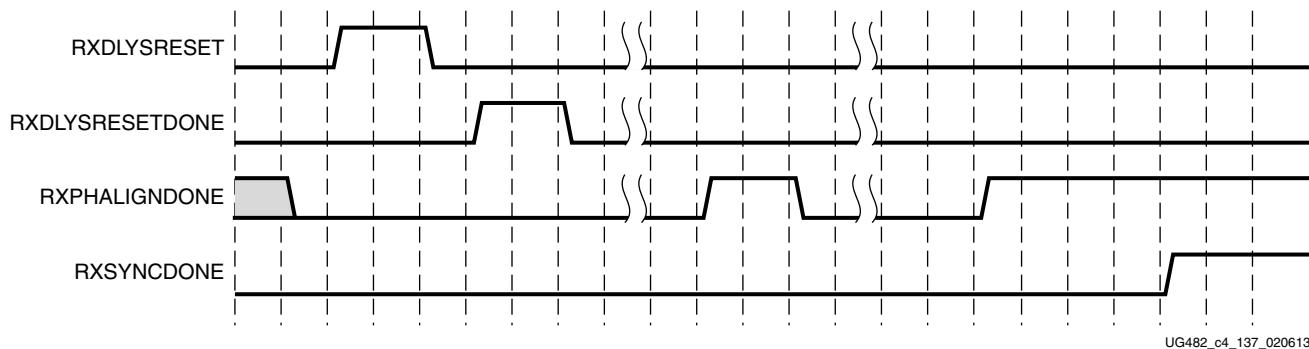
- RXSYNC\_MULTILANE = 0
- RXSYNC\_OVRD = 0

Set the ports as per Figure 4-36.



**Figure 4-36: RX Buffer Bypass—Single-Lane, Auto Mode Port Connection**

Figure 4-37 shows the required steps to perform the auto RX phase alignment and use the RX delay alignment to adjust RXUSRCLK to compensate for temperature and voltage variations.



**Figure 4-37: RX Buffer Bypass Example—Single-Lane Auto Mode**

Notes relevant to Figure 4-37:

1. The sequence of events in Figure 4-37 is not drawn to scale.
2. After conditions such as a GTP receiver reset or RX rate change, RX phase alignment must be performed to align XCLK and RXUSRCLK. Wait until exiting RXELECIDLE and RX CDR is locked before asserting RXDLYSRESET to start the RX phase and delay alignments. The assertion of RXDLYSRESET should be less than 50 ns.
3. Wait until RXDLYSRESETDONE is High. RXDLYSRESETDONE will stay asserted for a minimum of 100 ns.
4. When RXSYNCDONE is asserted, the alignment procedure is completed. This signal will remain asserted until the alignment procedure is re-initiated.
5. Upon the assertion of RXSYNCDONE, RXPHALIGNDONE indicates whether alignment is achieved and maintained.
6. RX delay alignment continues to adjust RXUSRCLK to compensate for temperature and voltage variations.

It is necessary to start the RX phase alignment after RX CDR is locked to ensure that the RX recovered clock and RXUSRCLK are stable and ready to be used for alignment. When the RX elastic buffer is bypassed, data received from the PMA can be distorted due to phase differences after conditions such as a GTP transceiver reset or rate change. If the received data evaluated at the fabric interface is invalid, the RX phase alignment needs to be repeated while the RX CDR is locked.

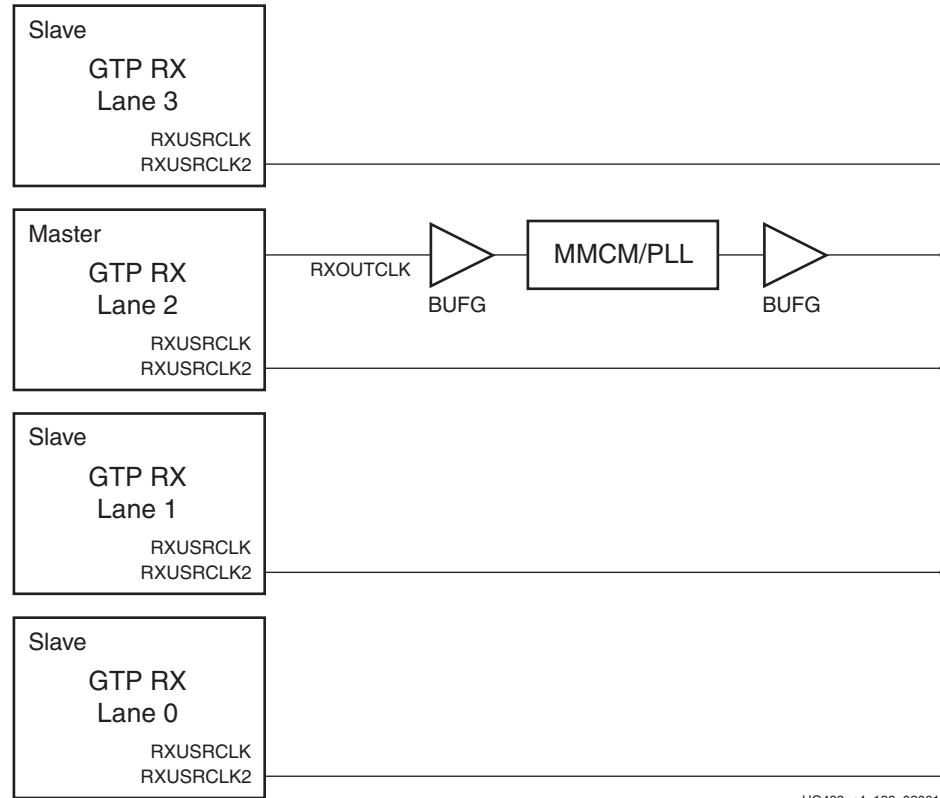
## Using RX Buffer Bypass in Multi-Lane Manual Mode

For GTP transceivers, phase alignment can be performed manually or automatically.

This section describes the steps required to perform the multi-lane RX buffer bypass alignment procedure manually:

- Master: In a multi-lane application, the buffer bypass master is the lane that is the source of RXOUTCLK.
- Slave: All the lanes that share the same RXUSRCLK/RXUSRCLK2, which is generated from the RXOUTCLK of the buffer bypass master.

[Figure 4-38](#) shows an example of buffer bypass master versus slave lanes.



[Figure 4-38: Example of RX Buffer Bypass Master versus Slave Lanes](#)

These GTP transceiver settings should be used to bypass the RX elastic buffer:

- RXBUF\_EN = FALSE
- RX\_XCLK\_SEL = RXUSR
- RXOUTCLKSEL = 010 to select the RX recovered clock as the source of RXOUTCLK
- RXDDIEN = 1

With the RX recovered clock selected, RXOUTCLK is to be used as the source of RXUSRCLK. The user must ensure that RXOUTCLK and the selected RX recovered clock are operating at the desired frequency. When the RX elastic buffer is bypassed, the RX phase alignment procedure must be performed after these conditions:

- Resetting or powering up the GTP transceiver receiver

- Resetting or powering up the PLL
- Changing the RX recovered clock source or frequency
- Changing the GTP transceiver RX line rate

Figure 4-39 shows the required steps to perform manual RX phase and delay alignment.



UG482\_c4\_39\_020713

**Figure 4-39: RX Phase and Delay Alignment in Manual Mode**

Notes relevant to Figure 4-39:

1. The sequence of events shown in Figure 4-39 is not drawn to scale.
2. M\_\* denotes ports related to the master lane.
3. S\_\* denotes ports related to the slave lane(s).
4. Set the RXSYNC\_OVRD attribute to 1'b1.
5. Set RXPHDLYRESET and RXDLYBYPASS to Low for all lanes.
6. Set RXPHALIGNEN and RXDDIEN to High for all lanes.
7. Assert RXDLYSRESET for all lanes. Hold this signal High until RXDLYSRESETDONE of the respective lane is asserted.
8. Deassert RXDLYSRESET for the lane in which the RXDLYSRESETDONE is asserted.
9. When RXDLYSRESET of all lanes are deasserted, assert RXPHALIGN for the master lane. Hold this signal High until the rising edge of RXPHALIGNDONE of the master lane is observed.
10. Deassert RXPHALIGN for the master lane.
11. Assert RXDLYEN for the master lane. This causes RXPHALIGNDONE to be deasserted.
12. Hold RXDLYEN for the master lane High until the rising edge of RXPHALIGNDONE of the master lane is observed.

13. Deassert RXDLYEN for the master lane.
14. Assert RXPHALIGN for all slave lane(s). Hold this signal High until the rising edge of RXPHALIGNDONE of the respective slave lane is observed.
15. Deassert RXPHALIGN for the slave lane in which the RXPHALIGNDONE is asserted.
16. When RXPHALIGN for all slave lane(s) are deasserted, assert RXDLYEN for the master lane. This causes RXPHALIGNDONE of the master lane to be deasserted.
17. Wait until RXPHALIGNDONE of the master lane reasserts. Phase and delay alignment for the multi-lane interface is complete. Continue to hold RXDLYEN for the master lane High to adjust RXUSRCLK to compensate for temperature and voltage variations.

In a multi-lane application, it is necessary to start the RX alignment procedure on the interface after RXELECIDLE is deasserted on any lane. The RX CDR of all lanes should be locked before starting the RX alignment procedure. This requirement is to ensure that the RX recovered clocks and RXUSRCLK are stable and ready before alignment.

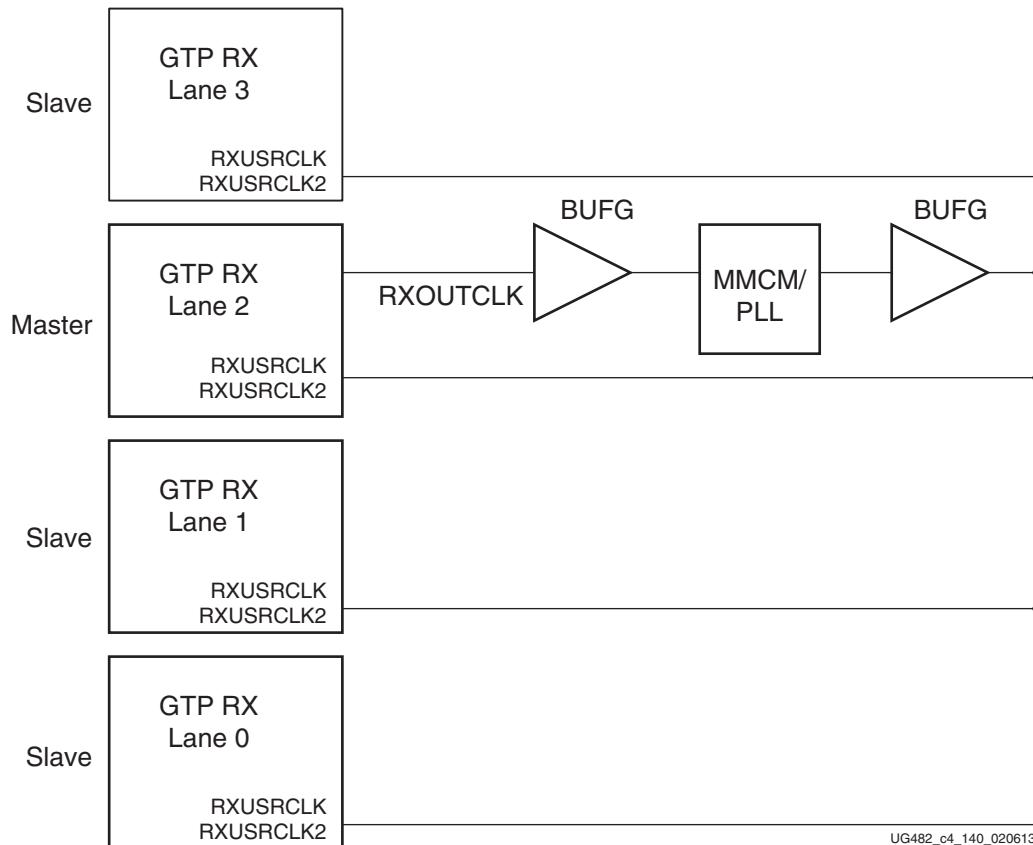
When the RX elastic buffer is bypassed, data received from the PMA might be distorted due to phase differences after conditions such as a GTP transceiver reset or rate change. If the received data evaluated at the fabric interface is invalid on any lane, the RX alignment procedure should be repeated for the interface after the RX CDR is locked on all lanes.

## Using RX Buffer Bypass in Multi-Lane Auto Mode

When a multi-lane application requires RX buffer bypass, phase alignment can be performed manually or automatically. This section describes the steps required to perform the multi-lane RX buffer bypass alignment procedure automatically:

- Master: In a multi-lane application, the buffer bypass master is the lane that is the source of RXOUTCLK.
- Slave: These are all the lanes that share the same RXUSRCLK/RXUSRCLK2, which is generated from the RXOUTCLK of the buffer bypass master.

Figure 4-40 shows an example of buffer bypass master versus slave lanes.



**Figure 4-40: Example of Buffer Bypass Master versus Slave Lanes**

These GTP transceiver settings should be used to bypass the RX buffer:

- RXBUF\_EN = FALSE.
- RX\_XCLK\_SEL = RXUSR.
- RXOUTCLKSEL = 010 to select the RX recovered clock as the source of RXOUTCLK.
- RXDDIEN = 1.

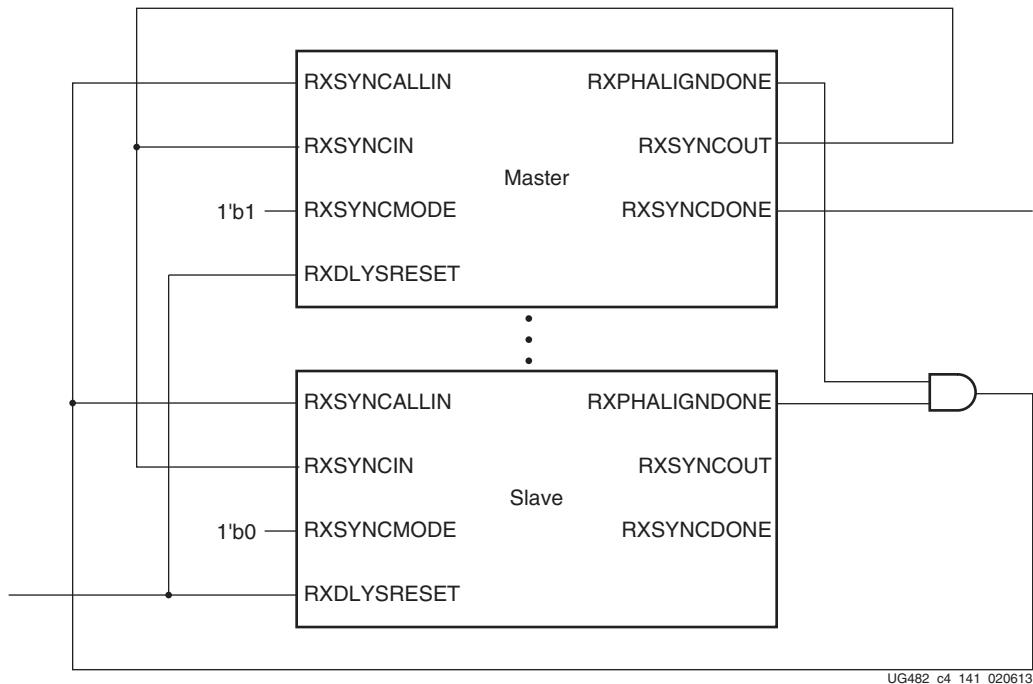
With the RX recovered clock selected, RXOUTCLK is to be used as the source of RXUSRCLK. The user must ensure that RXOUTCLK and the selected RX recovered clock are running and operating at the desire frequency. When the RX elastic buffer is bypassed, the RX phase alignment procedure must be performed after these conditions:

- Resetting or powering up the GTP receiver.
- Resetting or powering up the PLL.
- Changing the RX recovered clock source or frequency.
- Changing the GTP RX line rate.

To set up RX buffer bypass in multi-lane auto mode, the following attributes should be set:

- RXSYNC\_MULTILANE = 1
- RXSYNC\_OVRD = 0

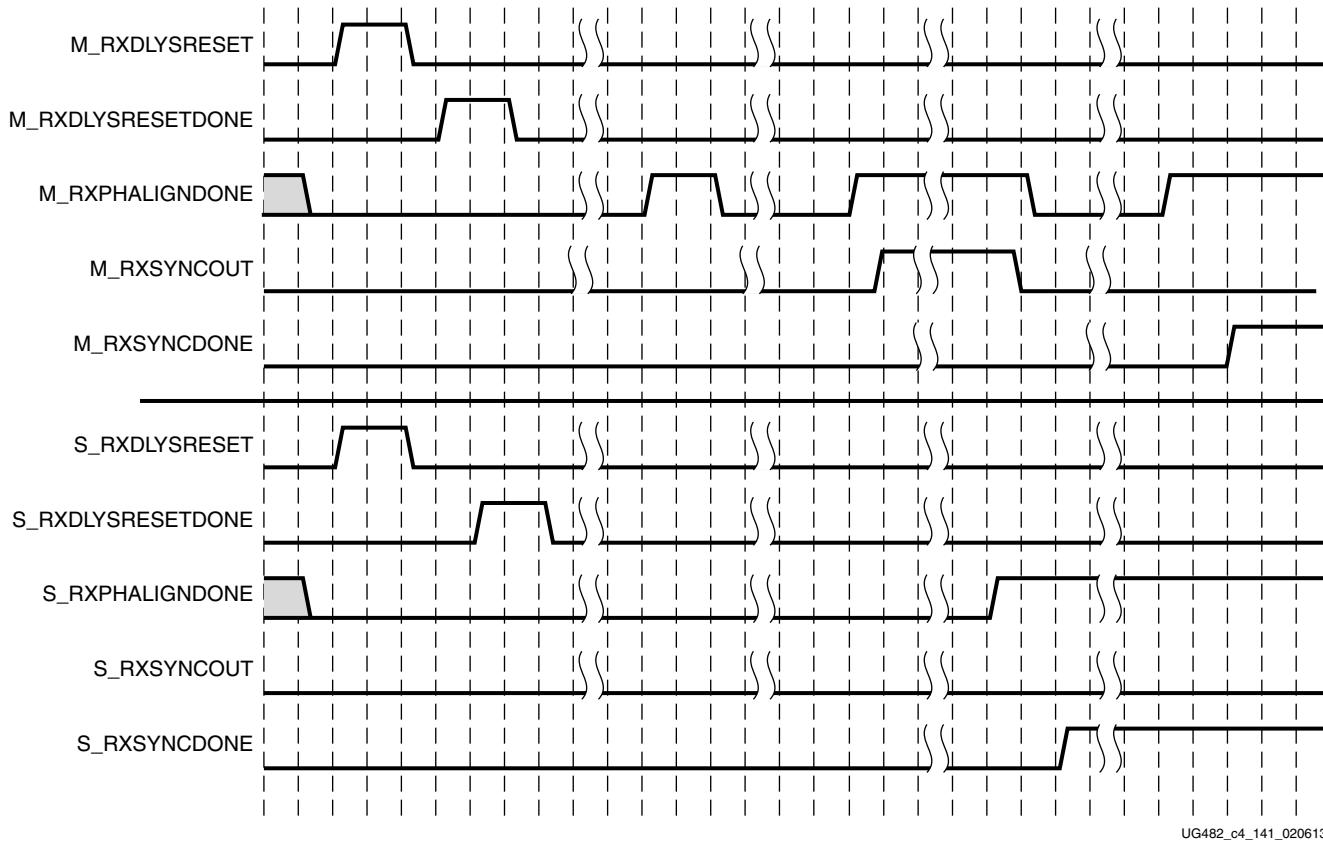
The ports should be set as shown in [Figure 4-41](#).



UG482\_c4\_141\_020613

Figure 4-41: RX Buffer Bypass—Multi-Lane Auto Mode Port Connection

Figure 4-42 shows the required steps to perform auto RX phase and delay alignment.



**Figure 4-42: RX Buffer Bypass Example—Multi-Lane Auto Mode**

Notes relevant to [Figure 4-42](#):

1. The sequence of events shown in [Figure 4-42](#) is not drawn to scale.
2. M\_\* denotes ports related to the master lane.
3. S\_\* denotes ports related to the slave lane(s).
4. After conditions such as a GTP receiver reset or RX rate change, RX phase alignment must be performed to align XCLK and RXUSRCLK. Wait until exiting RXELECIDLE and RX CDR is locked before asserting RXDLYSRESET to start the RX phase and delay alignments. The assertion of RXDLYSRESET should be less than 50 ns.
5. Wait until RXDLYSRESETDONE is High. RXDLYSRESETDONE will stay asserted for a minimum of 100 ns.
6. When RXSYNCDONE of the master lane is asserted, the alignment procedure is completed. This signal will remain asserted until alignment procedure is re-initiated.
7. Upon the assertion of RXSYNCDONE of the master lane, RXPHALIGNDONE of the master lane indicates whether alignment is achieved and maintained.
8. RX delay alignment continues to adjust RXUSRCLK to compensate for temperature and voltage variations.

In a multi-lane application, it is necessary to start the RX alignment procedure on the interface after RXELECIDLE is deasserted on any lane. RX CDR of all lanes needs to be locked before starting the RX alignment procedure. This requirement is to make sure the RX recovered clocks and RXUSRCLK are stable and ready before alignment.

It is necessary to start the RX phase alignment after RX CDR is locked to ensure that the RX recovered clock and RXUSRCLK are stable and ready to be used for alignment.

When the RX elastic buffer is bypassed, data received from the PMA can be distorted due to phase differences after conditions such as a GTP transceiver reset or rate change. If the received data evaluated at the fabric interface is invalid on any lane, the RX alignment procedure needs to be repeated for the interface after RX CDR is locked on all lanes.

## RX Elastic Buffer

### Functional Description

The GTP transceiver RX datapath has two internal parallel clock domains used in the PCS: The PMA parallel clock domain (XCLK) and the RXUSRCLK domain. To receive data, the PMA parallel rate must be sufficiently close to the RXUSRCLK rate, and all phase differences between the two domains must be resolved. [Figure 4-43](#) shows the two parallel clock domains: XCLK and RXUSRCLK.

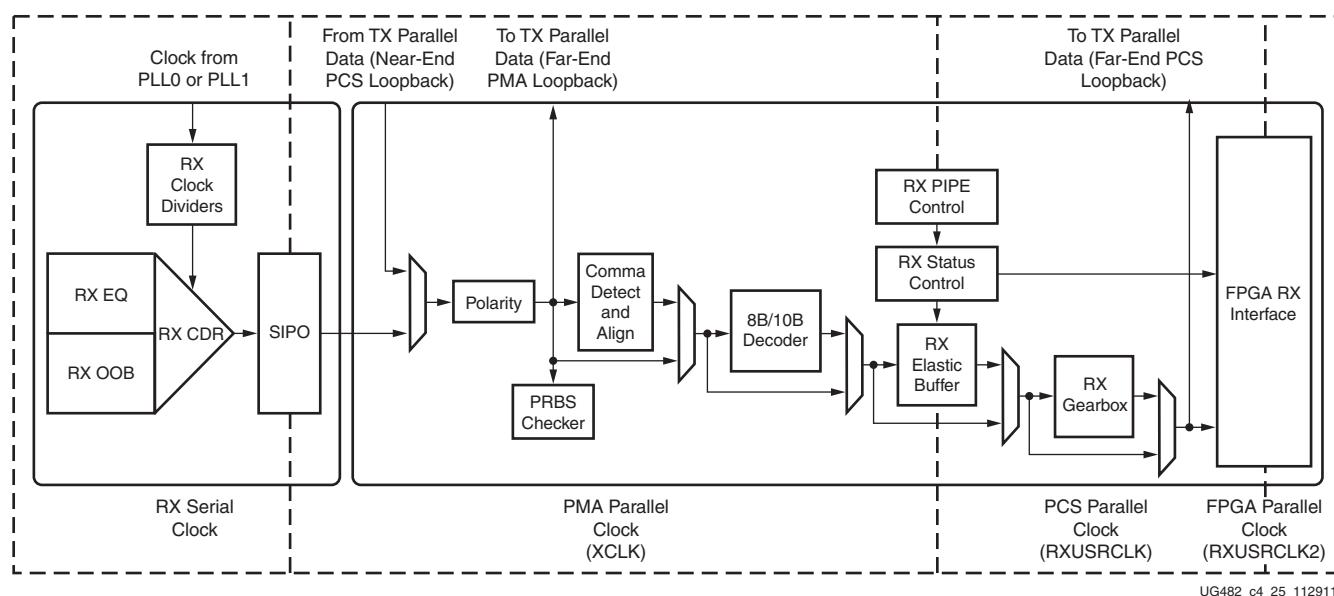


Figure 4-43: RX Clock Domains

The GTP transceiver includes an RX elastic buffer to resolve differences between the XCLK and RXUSRCLK domains. The phase of the two domains can also be matched by using the RX recovered clock from the transceiver to drive RXUSRCLK and adjusting its phase to match XCLK when the RX buffer is bypassed (see [RX Buffer Bypass, page 177](#)). All RX datapaths must use one of these approaches. The costs and benefits of each approach are shown in [Table 4-32](#).

**Table 4-32: RX Buffering versus Phase Alignment**

	<b>RX Elastic Buffer</b>	<b>RX Phase Alignment</b>
Ease of Use	The RX buffer is the recommended default to use when possible. It is robust and easier to operate.	Phase alignment is an advanced feature that requires extra logic and additional constraints on clock sources. RXOUTCLKSEL must select the RX recovered clock as the source of RXOUTCLK to drive RXUSRCLK.
Clocking Options	Can use RX recovered clock or local clock (with clock correction).	Must use the RX recovered clock.
Initialization	Works immediately.	Must wait for all clocks to stabilize before performing the RX phase and delay alignment procedure.
Latency	Buffer latency depends on features use, such as clock correction and channel bonding.	Lower deterministic latency.
Clock Correction and Channel Bonding	Required for clock correction and channel bonding.	Not performed inside the transceiver. Required to be implemented in user logic.

## Ports and Attributes

**Table 4-33** defines the RX buffer ports.**Table 4-33: RX Buffer Ports**

<b>Port</b>	<b>Dir</b>	<b>Clock Domain</b>	<b>Description</b>
RXBUFFRESET	In	Async	Resets and reinitializes the RX elastic buffer.
RXBUFFSTATUS[2:0]	Out	RXUSRCLK2	RX buffer status. 3'b000: Nominal condition. 3'b001: Number of bytes in the buffer are less than CLK_COR_MIN_LAT 3'b010: Number of bytes in the buffer are greater than CLK_COR_MAX_LAT 3'b101: RX elastic buffer underflow 3'b110: RX elastic buffer overflow

**Table 4-34** defines the RX buffer attributes.**Table 4-34: RX Buffer Attributes**

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
RXBUFF_EN	String	Use or bypass the RX elastic buffer. TRUE: Uses the RX elastic buffer (default). FALSE: Bypasses the RX elastic buffer (advanced feature).
RX_XCLK_SEL	String	Selects the clock source used to drive the RX parallel clock domain (XCLK). RXREC: Selects the RX recovered clock as the source of XCLK. Used when using the RX elastic buffer. RXUSR: Selects RXUSRCLK as the source of XCLK. Used when bypassing the RX elastic buffer.

Table 4-34: RX Buffer Attributes (Cont'd)

Attribute	Type	Description
RX_BUFFER_CFG	6-bit Binary	RX elastic buffer configuration. Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RX_DEFER_RESET_BUF_EN	String	Defer RX elastic buffer reset on comma realignment. The time deferred is controlled by RXBUF_EIDLE_HI_CNT.  TRUE: Enables deferral of RX elastic buffer reset on comma realignment.  FALSE: Disables deferral of RX elastic buffer reset on comma realignment.
RXBUF_ADDR_MODE	String	RX elastic buffer address mode.  FULL: Enables the RX elastic buffer for clock correction and channel bonding support.  FAST: Enables the RX elastic buffer for phase compensation without clock correction and channel bonding support. This mode is recommended for high line rates.
RXBUF_EIDLE_HI_CNT	4-bit Binary	Controls the timing of asserting the GTP transceiver internally generated RX elastic buffer reset on electrical idle when valid data is not present on the RXP/RXN serial lines. Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXBUF_EIDLE_LO_CNT	4-bit Binary	Controls the timing of deasserting the GTP transceiver internally generated RX elastic buffer reset on electrical idle when valid data is present on the RXP/RXN serial lines. Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.
RXBUF_RESET_ON_CB_CHANGE	String	GTP transceiver internally generated RX elastic buffer reset on channel bonding change.  TRUE: Enables auto RX elastic buffer reset on channel bonding change.  FALSE: Disables auto RX elastic buffer reset on channel bonding change.
RXBUF_RESET_ON_COMMALIGN	String	GTP transceiver internally generated RX elastic buffer reset on comma realignment.  TRUE: Enables auto RX elastic buffer reset on comma alignment.  FALSE: Disables auto RX elastic buffer reset on comma alignment.

Table 4-34: RX Buffer Attributes (Cont'd)

Attribute	Type	Description
RXBUT_RESET_ON_EIDLE	String	<p>GTP transceiver internally generated RX elastic buffer reset on electrical idle.</p> <p>TRUE: Enables auto reset of RX elastic buffer during an optional reset sequence of an electrical idle state as used in PCI Express operation.</p> <p>FALSE: Disables auto RX elastic buffer reset on electrical idle. This should be the default setting.</p> <p>Note: For channels with large attenuation (lossy channels typically exceeding 15 dB at Nyquist), it is recommended that RXBUF_RESET_ON_EIDLE should be set to FALSE because fast transitioning data patterns like the 101010 sequence in CJPAT/CJTPAT can accidentally trigger an electrical idle.</p>
RXBUT_RESET_ON_RATE_CHANGE	String	<p>GTP transceiver internally generated RX elastic buffer reset on rate change.</p> <p>TRUE: Enables auto RX elastic buffer reset on rate change.</p> <p>FALSE: Disables auto RX elastic buffer reset on rate change.</p>
RXBUT_THRESH_OVRD	String	<p>RX elastic buffer threshold override.</p> <p>TRUE: Use the RXBUF_THRESH_OVFLW and RXBUF_THRESH_UNDFLW attributes to set the RX elastic buffer overflow and underflow thresholds, respectively.</p> <p>FALSE: Automatically calculates the RX elastic buffer overflow and underflow thresholds. This is the recommended default setting.</p>
RXBUT_THRESH_OVFLW	Integer	<p>RX elastic buffer overflow threshold specified as the number of bytes. If the data latency through the RX elastic buffer is at or above this threshold, the buffer is considered to be in an overflow condition. Used when RXBUF_THRESH_OVRD = TRUE.</p> <p>Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.</p>
RXBUT_THRESH_UNDFLW	Integer	<p>RX elastic buffer underflow threshold specified as number of bytes. If the data latency through the RX elastic buffer is at or below this threshold, the buffer is consider to be in underflow condition. Used when RXBUF_THRESH_OVRD = TRUE.</p> <p>Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.</p>
RXBUTRESET_TIME	5-bit Binary	<p>RX elastic buffer reset time.</p> <p>Reserved. The recommended value from the 7 Series FPGAs Transceivers Wizard should be used.</p>

## Using the RX Elastic Buffer

These settings are used to enable the RX elastic buffer to resolve phase differences between the XCLK and RXUSRCLK domains:

- RXBUF\_EN = TRUE
- RX\_XCLK\_SEL = RXREC

The content of the RX elastic buffer becomes invalid if an RX elastic buffer overflow or underflow condition occurs. When any of these conditions occur, the RX elastic buffer should be reset and reinitialized by using GTRXRESET, RXPCSRESET, RXBUFRRESET, or the GTP transceiver internally generated RX elastic buffer reset. The internally generated RX elastic buffer reset can occur on channel bonding change, comma realignment, electrical idle, or rate change conditions.

The RX elastic buffer is also used for clock correction (see [RX Clock Correction](#)) and channel bonding (see [RX Channel Bonding, page 201](#)). Clock correction is used in cases where XCLK and RXUSRCLK are not frequency matched. [Table 4-35](#) lists common clock configurations and shows whether they require clock correction.

**Table 4-35: Common Clock Configurations**

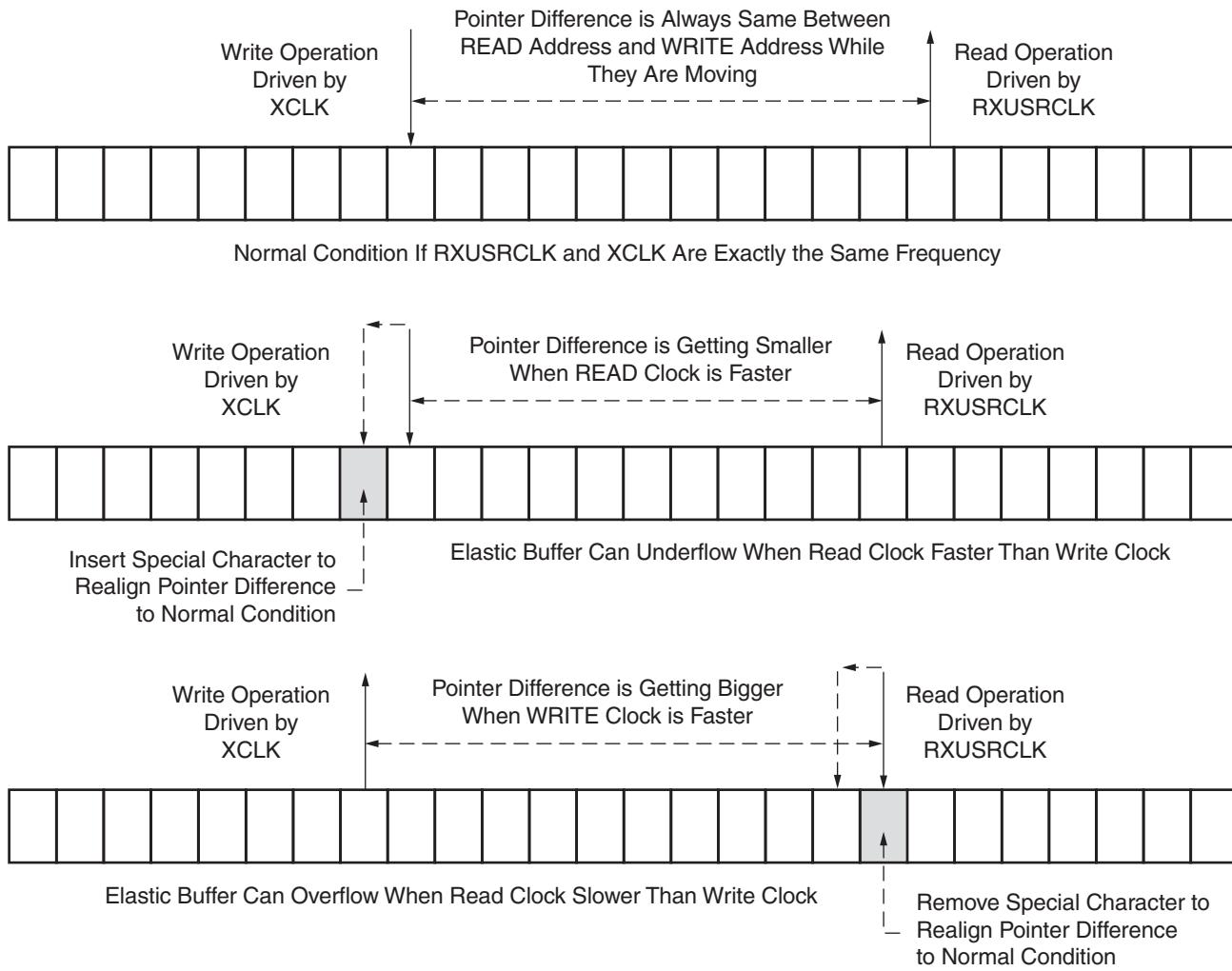
Types of Clocking	Require Clock Correction?
Synchronous system where both sides uses the reference clock from the same physical oscillator.	No
Asynchronous system when separate reference clocks are used and the GTP transceiver receiver uses an RX recovered clock.	No
Asynchronous system when separate reference clocks are used and the GTP transceiver receiver uses a local clock.	Yes

When the RX elastic buffer is used, the setting of CLK\_COR\_MIN\_LAT affects the latency through the buffer, regardless of whether clock correction is used.

## RX Clock Correction

### Functional Description

The RX elastic buffer is designed to bridge between two different clock domains, RXUSRCLK and XCLK, which is the recovered clock from CDR. Even if RXUSRCLK and XCLK are running at same clock frequency, there is always a small frequency difference. Because XCLK and RXUSRCLK are not exactly the same, the difference can be accumulated to cause the RX elastic buffer to eventually overflow or underflow unless it is corrected. To allow correction, each GTP transceiver TX periodically transmits one or more special characters that the GTP transceiver RX is allowed to remove or replicate in the RX elastic buffer as necessary. By removing characters when the RX elastic buffer is too full and replicating characters when the RX elastic buffer is too empty, the receiver can prevent overflow or underflow.



UG482\_c4\_26\_071612

Figure 4-44: Clock Correction Conceptual View

## Ports and Attributes

[Table 4-36](#) defines the ports required by RX clock correction functions.

**Table 4-36: RX Clock Correction Ports**

Port	Dir	Clock Domain	Description
RXBUFFRESET	In	Async	Resets the RX elastic buffer and related logic.
RXBUFFSTATUS[2:0]	Out	RXUSRCLK2	Indicates the status of the RX elastic buffer: 000: In nominal operating range where the buffer occupancy is within the CLK_COR_MIN_LAT and CLK_COR_MAX_LAT range 001: RX elastic buffer occupancy is less than CLK_COR_MIN_LAT 010: RX elastic buffer occupancy is greater than CLK_COR_MAX_LAT 101: RX elastic buffer underflow 110: RX elastic buffer overflow
RXCLKCORCNT[1:0]	Out	RXUSRCLK2	Reports the clock correction status of the RX elastic buffer when the first byte of a clock correction sequence is shown in RXDATA. 00: No clock correction 01: One sequence skipped 10: Two sequences skipped 11: One sequence added
RX8B10BEN	In	RXUSRCLK2	Active High to enable the 8B/10B decoder in the GTP transceiver RX. If 8B/10B decoding is enabled, RX_DATA_WIDTH must be a multiple of 10 (20, 40). If 8B/10B decoding is not enabled, RX_DATA_WIDTH must be a multiple of 8 (16, 32).

[Table 4-37](#) defines the attributes required by RX channel bonding.

**Table 4-37: RX Clock Correction Attributes**

Attribute	Type	Description
CBCC_DATA_SOURCE_SEL	String	This attribute is used together with RX8B10BEN to select the data source for clock correction and channel bonding. When RX8B10BEN is High, CBCC_DATA_SOURCE_SEL = DECODED, the clock correction sequence matches the data decoded after the 8B/10B decoder. CBCC_DATA_SOURCE_SEL = ENCODED, the clock correction sequence matches the raw data from the comma detection and realignment block before the 8B/10B decoder. When RX8B10BEN is Low, CBCC_DATA_SOURCE_SEL = DECODED is not supported. CBCC_DATA_SOURCE_SEL = ENCODED, the clock correction sequence matches the raw data from the comma detection and realignment block.

Table 4-37: RX Clock Correction Attributes (Cont'd)

Attribute	Type	Description
CLK_CORRECT_USE	String	<p>Set TRUE to enable the clock correction function. Set FALSE to disable the clock correction function.</p> <p>These attributes need to be set while clock correction disabled:</p> <ul style="list-style-type: none"> <li>CLK_COR_SEQ_1_1 = 10'b0100000000</li> <li>CLK_COR_SEQ_2_1 = 10'b0100000000</li> <li>CLK_COR_SEQ_1_ENABLE = 4'b1111</li> <li>CLK_COR_SEQ_2_ENABLE = 4'b1111</li> </ul>
CLK_COR_KEEP_IDLE	String	<p>Set TRUE to keep at least one clock correction sequence in the data stream for every continuous stream of clock correction sequences received.</p> <p>Set FALSE to remove all clock correction sequences from the byte stream if needed to recenter the RX elastic buffer range.</p>
CLK_COR_MAX_LAT	Integer	<p>Specifies the maximum RX elastic buffer latency. If the RX elastic buffer exceeds CLK_COR_MAX_LAT, the clock correction circuit removes incoming clock correction sequences to prevent overflow.</p> <p>The 7 Series FPGAs Transceivers Wizard chooses an optimal CLK_COR_MAX_LAT value based on application requirements. The value selected by the Wizard must be followed to maintain optimal performance and must not be overridden.</p>
CLK_COR_MIN_LAT	Integer	<p>Specifies the minimum RX elastic buffer latency. If the RX elastic buffer drops below CLK_COR_MIN_LAT, the clock correction circuit replicates incoming clock correction sequences to prevent underflow.</p> <p>When the RX elastic buffer is reset, its pointers are set so that there are CLK_COR_MIN_LAT unread (and uninitialized) data bytes in the buffer.</p> <p>The 7 Series FPGAs Transceivers Wizard chooses a CLK_COR_MIN_LAT value based on application requirements. The value selected by the Wizard must be followed to maintain optimal performance and must not be overridden.</p>
CLK_COR_PRECEDENCE	String	<p>Determines whether clock correction or channel bonding takes precedence when both operations are triggered at the same time.</p> <p>TRUE: Clock correction takes precedence over channel bonding if there is opportunity for both</p> <p>FALSE: Channel bonding takes precedence over clock correction if there is opportunity for both</p>
CLK_COR_REPEAT_WAIT	Integer	<p>This attribute specifies the minimum number of RXUSRCLK cycles between two successive clock corrections being placed. If this attribute is 0, no limit is placed on how frequently the clock correction character can be placed.</p> <p>Valid values for this attribute range from 0 to 31.</p>
CLK_COR_SEQ_LEN	Integer	<p>Defines the length of the sequence in bytes that has to match to detect opportunities for clock correction. This attribute also defines the size of the adjustment (number of bytes repeated or skipped) in a clock correction.</p> <p>Valid lengths are 1, 2, and 4 bytes.</p>

Table 4-37: RX Clock Correction Attributes (Cont'd)

Attribute	Type	Description
CLK_COR_SEQ_1_ENABLE	4-bit Binary	<p>Mask enable bit for the first clock correction sequence.</p> <p>CLK_FOR_SEQ_1_ENABLE[0] is the mask bit for CLK_COR_SEQ_1_1.</p> <p>CLK_FOR_SEQ_1_ENABLE[1] is the mask bit for CLK_COR_SEQ_1_2.</p> <p>CLK_FOR_SEQ_1_ENABLE[2] is the mask bit for CLK_COR_SEQ_1_3.</p> <p>CLK_FOR_SEQ_1_ENABLE[3] is the mask bit for CLK_COR_SEQ_1_4.</p> <p>When CLK_FOR_SEQ_1_ENABLE[*] is 0, the corresponding CLK_COR_SEQ_1_* is either considered as a don't care or is matched automatically without a comparison.</p> <p>When CLK_FOR_SEQ_1_ENABLE[*] is 1, the corresponding CLK_COR_SEQ_1_* is compared for a match.</p>
CLK_COR_SEQ_1_1	10-bit Binary	First clock correction sequence 1 to be compared when CLK_FOR_SEQ_1_ENABLE[0] = 1.
CLK_COR_SEQ_1_2	10-bit Binary	First clock correction sequence 2 to be compared when CLK_FOR_SEQ_1_ENABLE[1] = 1.
CLK_COR_SEQ_1_3	10-bit Binary	First clock correction sequence 3 to be compared when CLK_FOR_SEQ_1_ENABLE[2] = 1.
CLK_COR_SEQ_1_4	10-bit Binary	First clock correction sequence 4 to be compared when CLK_FOR_SEQ_1_ENABLE[3] = 1.
CLK_COR_SEQ_2_USE	String	Set to TRUE if the second clock correction sequence (CLK_COR_SEQ_2_*) is used in addition to the CLK_COR_SEQ_1_* that is always used.
CLK_COR_SEQ_2_ENABLE	4-bit Binary	<p>Mask enable bit for the second clock correction sequence.</p> <p>CLK_FOR_SEQ_2_ENABLE[0] is the mask bit for CLK_COR_SEQ_2_1.</p> <p>CLK_FOR_SEQ_2_ENABLE[1] is the mask bit for CLK_COR_SEQ_2_2.</p> <p>CLK_FOR_SEQ_2_ENABLE[2] is the mask bit for CLK_COR_SEQ_2_3.</p> <p>CLK_FOR_SEQ_2_ENABLE[3] is the mask bit for CLK_COR_SEQ_2_4.</p> <p>When CLK_FOR_SEQ_2_ENABLE[*] is 0, the corresponding CLK_COR_SEQ_2_* is either considered as a don't care or is matched automatically without a comparison.</p> <p>When CLK_FOR_SEQ_2_ENABLE[*] is 1, the corresponding CLK_COR_SEQ_2_* is compared for a match.</p>
CLK_COR_SEQ_2_1	10-bit Binary	Second clock correction sequence 1 to be compared when CLK_FOR_SEQ_2_ENABLE[0] = 1
CLK_COR_SEQ_2_2	10-bit Binary	Second clock correction sequence 2 to be compared when CLK_FOR_SEQ_2_ENABLE[1] = 1
CLK_COR_SEQ_2_3	10-bit Binary	Second clock correction sequence 3 to be compared when CLK_FOR_SEQ_2_ENABLE[2] = 1
CLK_COR_SEQ_2_4	10-bit Binary	Second clock correction sequence 4 to be compared when CLK_FOR_SEQ_2_ENABLE[3] = 1

Table 4-37: RX Clock Correction Attributes (Cont'd)

Attribute	Type	Description
RX_DATA_WIDTH	Integer	Sets the bit width of the RXDATA port. When 8B/10B encoding is enabled, RX_DATA_WIDTH must be set to 20 or 40. Valid settings are 16, 20, 32, and 40. See <a href="#">Interface Width Configuration, page 219</a> for more details.
RX_DISPERR_SEQ_MATCH	String	Specifies whether the disparity error status of a decoded byte must match the indicator in the channel bonding and clock correction sequence. TRUE: The disparity error status must be matched. FALSE: The disparity error status is ignored.
ALIGN_COMMA_WORD	Integer	This attribute controls the alignment of detected commas within a multi-byte datapath. 1: Align the comma to either of the 2 bytes for a 2-byte interface and any of the 4 bytes for a 4-byte interface. The comma can be aligned to either the even bytes or the odd bytes of the RXDATA output. 2: Align the comma to the even bytes only. The aligned comma is guaranteed to be aligned to even bytes RXDATA[9:0] for a 2-byte interface and RXDATA[9:0]/RXDATA[29:20] for a 4-byte interface. Refer to <a href="#">Figure 4-30</a> for comma alignment boundaries that are allowed for the different ALIGN_COMMA_WORD and RX_DATA_WIDTH settings. Protocols that send commas in even and odd positions must set ALIGN_COMMA_WORD to 1.

## Using RX Clock Correction

The user must follow the steps described in this section to use the receiver's clock correction feature.

### Enabling Clock Correction

Each GTP transceiver includes a clock correction circuit that performs clock correction by controlling the pointers of the RX elastic buffer. To use clock correction, RXBUF\_EN is set to TRUE to turn on the RX elastic buffer, and CLK\_CORRECT\_USE is set to TRUE to turn on the clock correction circuit.

Clock correction is triggered when the RX elastic buffer latency is too high or too low, and the clock correction circuit detects a match sequence. To use clock correction, the clock correction circuit must be configured to set these items:

- RX elastic buffer limits
- Clock correction sequence

### Setting RX Elastic Buffer Limits

The RX elastic buffer limits are set using CLK\_COR\_MIN\_LAT (minimum latency) and CLK\_COR\_MAX\_LAT (maximum latency). When the number of bytes in the RX elastic buffer drops below CLK\_COR\_MIN\_LAT, the clock correction circuit writes an additional CLK\_COR\_SEQ\_LEN byte from the first clock correction sequence it matches to prevent

buffer underflow. Similarly, when the number of bytes in the RX elastic buffer exceeds CLK\_COR\_MAX\_LAT, the clock correction circuit deletes CLK\_COR\_SEQ\_LEN bytes from the first clock correction sequence it matches, starting with the first byte of the sequence. The 7 Series FPGAs Transceivers Wizard chooses an optimal setting for CLK\_COR\_MIN\_LAT and CLK\_COR\_MAX\_LAT based on application requirements. The values selected by the Wizard must be followed to maintain optimal performance and must not be overridden.

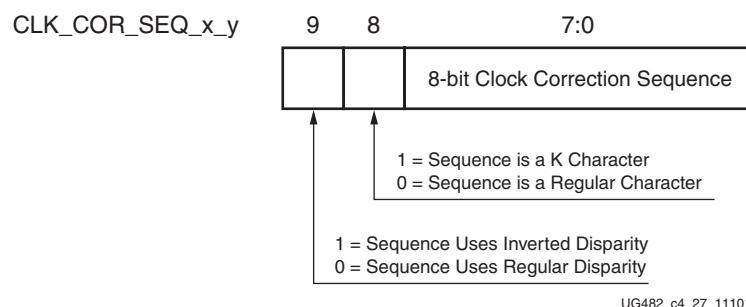
## Setting Clock Correction Sequences

The clock correction sequences are programmed using the CLK\_COR\_SEQ\_1\_\* attributes and CLK\_COR\_SEQ\_LEN. Each CLK\_COR\_SEQ\_1\_\* attribute corresponds to one subsequence in clock correction sequence 1. CLK\_COR\_SEQ\_LEN is used to set the number of subsequences to be matched. If the 40-bit or 20-bit internal datapaths are used, the clock correction circuit matches all 10 bits of each subsequence. If the 16-bit or 32-bit internal datapaths are used, only the right-most eight bits of each subsequence are used.

A second, alternate clock correction sequence can be activated by setting CLK\_COR\_SEQ\_2\_USE to TRUE. The first and second sequences share length settings, but use different subsequence values for matching. Set the CLK\_COR\_SEQ\_2\_\* attributes to define the subsequence values for the second sequence.

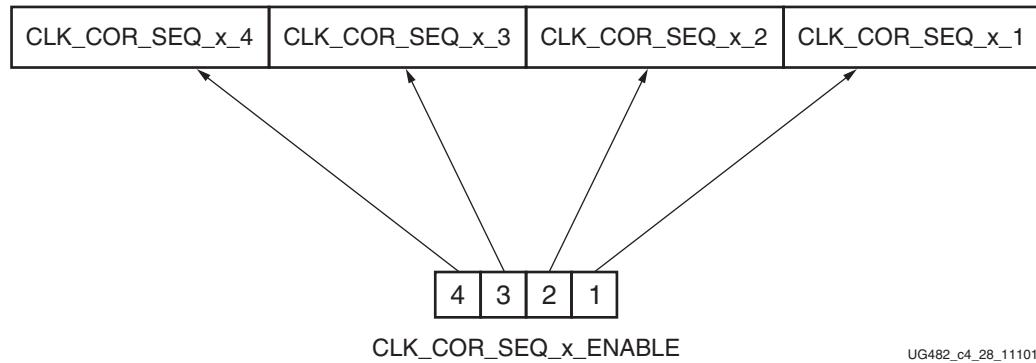
When using 8B/10B decoding (RX8B10BEN is High), CBCC\_DATA\_SOURCE\_SEL is set to DECODED to search the output of the 8B/10B decoder for sequence matches instead of non-decoded data. This allows the circuit to look for 8-bit values with either positive or negative disparity, and to distinguish K characters from regular characters (see [TX 8B/10B Encoder, page 85](#) and [RX 8B/10B Decoder, page 173](#) for details). [Figure 4-45](#) shows how to set a clock correction sequence byte when RX8B10BEN is High and CBCC\_DATA\_SOURCE\_SEL is set to DECODED.

When CBCC\_DATA\_SOURCE\_SEL is set to ENCODED, the sequence must exactly match incoming raw data. When RX\_DISPERR\_SEQ\_MATCH is set to FALSE, CLK\_COR\_SEQ\_x\_y[9] is not used for matching.



*Figure 4-45: Clock Correction Subsequence Settings with CBCC\_DATA\_SOURCE\_SEL = DECODED*

Some protocols use clock correction sequences with don't care subsequences. The clock correction circuit can be programmed to recognize these sequences using CLK\_COR\_SEQ\_1\_ENABLE and CLK\_COR\_SEQ\_2\_ENABLE. When the enable bit for a sequence is Low, that byte is considered matched no matter what the value is. [Figure 4-46](#) shows the mapping between the clock correction sequences and the clock correction sequence enable bits.



UG482\_c4\_28\_111011

**Figure 4-46: Clock Correction Sequence Mapping**

To preserve comma alignment through the elastic buffer, CLK\_COR\_SEQ\_LEN and ALIGN\_COMMA\_WORD must be selected such that they comply with [Table 4-38](#).

**Table 4-38: Valid ALIGN\_COMMA\_WORD/CLK\_COR\_SEQ\_LEN Combinations**

ALIGN_COMMA_WORD	CLK_COR_SEQ_LEN
1	1, 2, 4
2	2, 4

## Clock Correction Options

CLK\_COR\_REPEAT\_WAIT is used to control the clock correction frequency. This value is set to the minimum number of RXUSRCLK cycles required between clock correction events. This attribute is set to 0 to allow clock correction to occur at any time. Some protocols allow clock correction to occur at any time, but require that if the clock correction circuit removes sequences, at least one sequence stays in the stream. For protocols with this requirement, CLK\_COR\_KEEP\_IDLE is set to TRUE.

## Monitoring Clock Correction

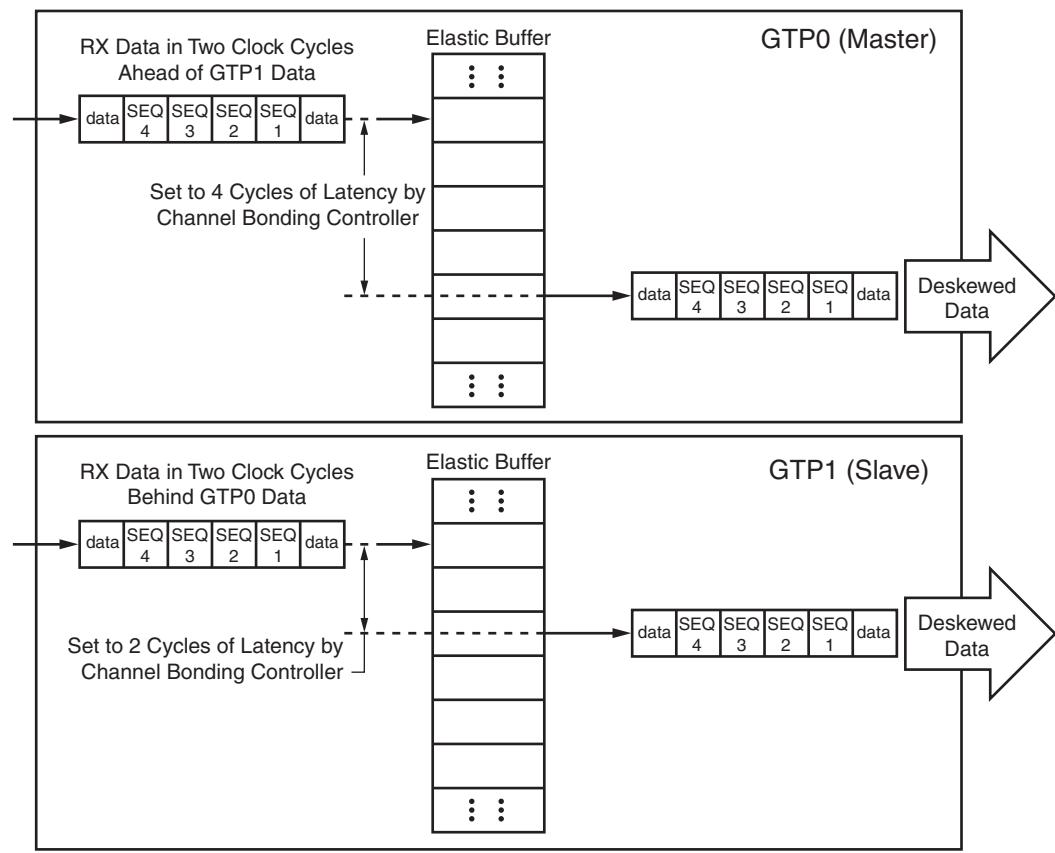
The clock correction circuit can be monitored using the RXCLKCORCNT and RXBUFSTATUS ports. The RXCLKCORCNT entry in [Table 4-36](#) shows how to decode the values of RXCLKCORCNT to determine the status of the clock correction circuit. The RXBUFSTATUS entry in [Table 4-36](#) shows how to decode the values of RXBUFSTATUS to determine how full the RX elastic buffer is.

# RX Channel Bonding

## Functional Description

Protocols such as XAUI and PCI Express combine multiple serial transceiver connections to create a single higher throughput channel. Each serial transceiver connection is called one lane. Unless each of the serial connections is exactly the same length, skew between the lanes can cause data to be transmitted at the same time but arrive at different times. Channel bonding cancels out the skew between GTP transceiver lanes by using the RX elastic buffer as a variable latency block. Channel bonding is also called channel deskew or lane-to-lane deskew. GTP transceiver transmitters used for a bonded channel all transmit a channel bonding character (or a sequence of characters) simultaneously. When the

sequence is received, the GTP transceiver receiver can determine the skew between each lane and adjust the latency of RX elastic buffers, so that data is presented without skew at the RX fabric user interface.



UG482\_c4\_29\_111011

**Figure 4-47: Channel Bonding Conceptual View**

RX channel bonding supports 8B/10B encoded data but does not support these encoded data types:

- 64B/66B
- 64B/67B
- 128B/130B
- Scrambled data

## Ports and Attributes

**Table 4-39** defines the ports required by RX channel bonding functions.

**Table 4-39: RX Channel Bonding Ports**

Port	Dir	Clock Domain	Description
RXCHANBONDSEQ	Out	RXUSRCLK2	This port goes High when RXDATA contains the start of a channel bonding sequence.
RXCHANISALIGNED	Out	RXUSRCLK2	This signal from the RX elastic buffer goes High to indicate that the channel is properly aligned with the master transceiver according to observed channel bonding sequences in the data stream. This signal goes Low if an unaligned channel bonding sequence is detected, indicating that channel alignment was lost.
RXCHANREALIGN	Out	RXUSRCLK2	This signal from the RX elastic buffer is held High for at least one cycle when the receiver has changed the alignment between this transceiver and the master.
RXCHBONDI[3:0]	In	RXUSRCLK	Channel bonding control ports used by slaves only. These ports are used to receive channel bonding and clock correction control information from master GTP transceiver RXCHBONDO ports or from daisy-chained slave GTP transceiver RXCHBONDO ports, which are concatenated from the master GTP transceiver.
RXCHBONDO[3:0]	Out	RXUSRCLK	Channel bonding control ports used to propagate channel bonding and clock correction information to the slave GTP transceiver from the master or a daisy-chained slave concatenated from the master. The master RXCHBONDO can be tied to one or multiple slave RXCHBONDI ports. The slave RXCHBONDO should be tied to the next level slave RXCHBONDI to form a daisy chain and pass information from the master to each slave.
RXCHBONDLEVEL[2:0]	In	RXUSRCLK2	Indicates the amount of internal pipelining used for the RX elastic buffer control signals. A higher value permits more daisy chaining of RXCHBONDO and RXCHBONDI to ease placement and routing constraints. To minimize required latency through the RX elastic buffer, CHAN_BOND_LEVEL in the master is set to the smallest value possible for the required amount of daisy-chaining.
RXCHBONDMASTER	In	RXUSRCLK2	Indicates that the transceiver is the master for channel bonding. Its RXCHBONDO port directly drives the RXCHBONDI ports on one or more slave transceivers. This port cannot be driven High at the same time as RXCHBONDLSLAVE.

Table 4-39: RX Channel Bonding Ports (Cont'd)

Port	Dir	Clock Domain	Description
RXCHBOND_SLAVE	In	RXUSRCLK2	Indicates that this transceiver is a slave for channel bonding. Its RXCHBONDI port is directly driven by the RXCHBONDO port of another slave or master transceiver. If its RXCHBONDOLEVEL[2:0] setting is greater than 0, its RXCHBONDO port can directly drive the RXCHBONDI ports on one or more other slave transceivers. This port cannot be driven High at the same time as RXCHBONDMASTER.
RXCHBONDEN	In	RXUSRCLK2	This port enables channel bonding (from the FPGA logic to both the master and slaves).

Table 4-40 defines the attributes required by RX channel bonding.

Table 4-40: RX Channel Bonding Attributes

Attribute	Type	Description
CHAN_BOND_MAX_SKEW	Integer	This attribute controls the number of USRCLK cycles that the master waits before ordering the slaves to execute channel bonding. This attribute determines the maximum skew that can be handled by channel bonding. It must always be less than one-half the minimum distance (in bytes or 10-bit codes) between channel bonding sequences. Valid values range from 1 to 14.
CHAN_BOND_KEEP_ALIGN	String	Allows preservation of ALIGN characters during channel bonding for PCI Express.
CHAN_BOND_SEQ_1_1 CHAN_BOND_SEQ_1_2 CHAN_BOND_SEQ_1_3 CHAN_BOND_SEQ_1_4	10-bit Binary	The CHAN_BOND_SEQ_1 attributes are used in conjunction with CHAN_BOND_SEQ_1_ENABLE to define channel bonding sequence 1. Each subsequence is 10 bits long. The rules for setting the subsequences depend on RX_DATA_WIDTH and CBCC_DATA_SOURCE_SEL.
CHAN_BOND_SEQ_1_ENABLE	4-bit Binary	Not all subsequences need to be used. CHAN_BOND_SEQ_LEN determines how much of the sequence is used for a match. If CHAN_BOND_SEQ_LEN = 1, only CHAN_BOND_SEQ_1_1 is used. CHAN_BOND_SEQ_1_ENABLE can be used to make parts of the sequence don't care. If CHAN_BOND_SEQ_1_ENABLE[k] is 0, CHAN_BOND_SEQ_1_k is a don't-care subsequence and is always considered to be a match.

Table 4-40: RX Channel Bonding Attributes (*Cont'd*)

Attribute	Type	Description
CHAN_BOND_SEQ_2_1 CHAN_BOND_SEQ_2_2 CHAN_BOND_SEQ_2_3 CHAN_BOND_SEQ_2_4	10-bit Binary	The CHAN_BOND_SEQ_2 attributes are used in conjunction with CHAN_BOND_SEQ_2_ENABLE to define the second channel bonding sequence. When CHAN_BOND_SEQ_2_USE is TRUE, the second sequence is used as an alternate sequence to trigger channel bonding.
CHAN_BOND_SEQ_2_ENABLE	4-bit Binary	<p>Each subsequence is 10 bits long. The rules for setting the subsequence depend on RX_DATA_WIDTH and CBCC_DATA_SOURCE_SEL.</p> <p>Not all subsequences need to be used. CHAN_BOND_SEQ_LEN determines how many of the subsequences are used for a match. If CHAN_BOND_SEQ_LEN = 1, only CHAN_BOND_SEQ_2_1 is used.</p> <p>CHAN_BOND_SEQ_2_ENABLE can be used to make parts of the sequence don't care. If CHAN_BOND_SEQ_2_ENABLE[k] is 0, CHAN_BOND_SEQ_2_k is a don't-care subsequence and is always considered to be a match.</p>
CHAN_BOND_SEQ_2_USE	String	Determines if the two-channel bonding sequence is to be used. TRUE: Channel bonding can be triggered by channel bonding sequence 1 or 2. FALSE: Channel bonding is only triggered by sequence 1.
CHAN_BOND_SEQ_LEN	Integer	Defines the length in bytes of the channel bonding sequence that the GTP transceiver has to match to find skew. Valid lengths are 1, 2, and 4 bytes.
CBCC_DATA_SOURCE_SEL	String	This attribute is used to select the data source for clock correction and channel bonding. When set to DECODED, selects data from the 8B/10B decoder when RX8B10BEN is High. When set to ENCODED, selects data from the comma detection and realignment block.
FTS_DESKEW_SEQ_ENABLE	4-bit Binary	<p>Enable mask for FTS_LANE_DESKEW_CFG.</p> <p>FTS_DESKEW_SEQ_ENABLE[0] is for FTS_LANE_DESKEW_CFG[0]</p> <p>FTS_DESKEW_SEQ_ENABLE[1] is for FTS_LANE_DESKEW_CFG[1]</p> <p>FTS_DESKEW_SEQ_ENABLE[2] is for FTS_LANE_DESKEW_CFG[2]</p> <p>FTS_DESKEW_SEQ_ENABLE[3] is for FTS_LANE_DESKEW_CFG[3]</p> <p>The default value is 1111.</p>

Table 4-40: RX Channel Bonding Attributes (*Cont'd*)

Attribute	Type	Description
FTS_LANE_DESKEW_CFG	4-bit Binary	<ul style="list-style-type: none"> <li>Bit 3: This bit is set to 1'b1 on a slave to freeze the alignment to prevent spurious misalignments or modified alignments that can occur following slip-4, snap-4, or clock correction when good channel alignment is still maintained. This bit is set to 1'b0 on a slave to unfreeze the alignment.</li> <li>Bit 2: Specifies whether a “master” channel doing FTS lane deskew that just reached the end of an FTS OS in its lookahead control logic inhibits its own generation of clock correction commands for a brief time. The purpose is to prevent clock correction commands from interfering with operation of the slave’s slip-4 and snap-4 logic. The logic guarantees that clock correction can still occur if a full SKP OS is present.</li> <li>Bit 1: Specifies whether a “slave” channel doing FTS lane deskew is permitted (1'b1) or inhibited (1'b0) from performing an immediate backward alignment adjustment by four bytes (slip-4) if the slave is found to have reached the SKP OS following FTS before the master has.</li> <li>Bit 0: Specifies whether a “slave” channel doing FTS lane deskew is permitted (1'b1) or inhibited (1'b0) from performing an immediate forward alignment adjustment by four bytes (snap-4) if the master is found to have reached the SKP OS following FTS before the slave has.</li> </ul>
FTS_LANE_DESKEW_EN	String	This attribute is set to TRUE to enable channel bonding logic for FTS lane deskew. FTS lane deskew is separate from the standard algorithm using channel bonding sequences 1 and 2, and it operates in parallel with the standard algorithm. FTS lane deskew operates only in two-byte mode.
PCS_PCIE_EN	String	This attribute is set to TRUE when the GTP transceiver is used for PCI Express and set to FALSE for all other protocols. The channel bonding function requires this attribute together with TXCHARDISPMODE and TXCHARDISPVAL to support PIPE encode and FTS lane deskew. It also works together with TXELECIDLE to match a shorter sequence from reusing prior channel bonding information after the GTP transceiver returns from electrical idle.
RX_DATA_WIDTH	Integer	Sets the bit width of the RXDATA port. When 8B/10B encoding is enabled, RX_DATA_WIDTH must be set to 20 or 40. Valid settings are 16, 20, 32, and 40. See <a href="#">Interface Width Configuration, page 219</a> for more details.
RX_DISPERR_SEQ_MATCH	String	Specifies whether the disparity error status of a decoded byte must match the indicator in the channel bonding and clock correction sequence. TRUE: The disparity error must be matched. FALSE: The disparity error status is ignored.

## Using RX Channel Bonding

The user must follow the steps described below to use the receiver’s channel bonding feature.

## Enabling Channel Bonding

Each GTP transceiver includes a circuit that performs channel bonding by controlling the pointers of the RX elastic buffer. Because channel bonding requires the use of the RX buffer, the RXBUF\_EN attribute must be set to TRUE.

Each GTP transceiver has a channel bonding circuit. Configuring a GTP transceiver for channel bonding requires these steps:

1. Set the channel bonding mode for each GTP transceiver.
2. Tie the RXCHBONDMASTER of the master transceiver High.
3. Tie the RXCHBONDSSLAVE of the slave transceiver(s) High.
4. Connect the channel bonding port from the master to each slave, either directly or by daisy chaining.
5. Set the channel bonding sequence and detection parameters.

## Channel Bonding Mode

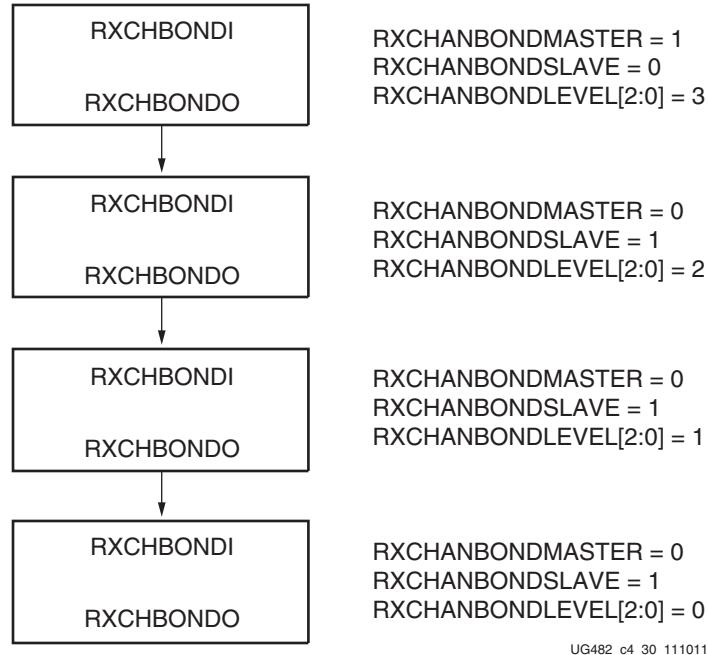
The channel bonding mode for each GTP transceiver determines whether channel bonding is active and whether the GTP transceiver is the master or a slave. Each set of channel bonded GTP transceivers must have one master and any number of slaves. To turn on channel bonding for a group of GTP transceivers, one transceiver is set to master. The remaining GTP transceivers in the group are set to slaves.

## Connecting Channel Bonding Ports

The channel bonding operation requires connecting the master GTP transceiver RXCHBONDO port to the RXCHBONDI port of all slaves in the group. Only GTP transceivers belonging to the same column can be channel bonded together. A direct connection is required for adjacent GTP transceivers. To directly connect a master to a slave:

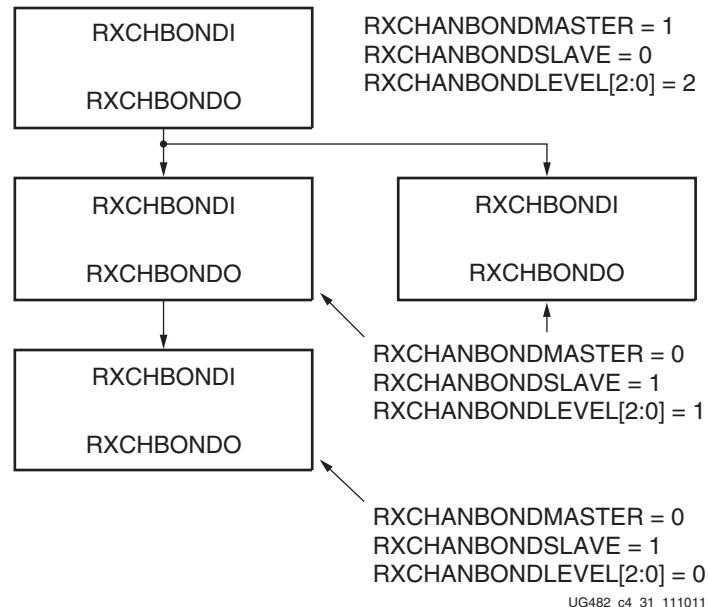
1. Connect the RXCHBONDO port of the master to the RXCHBONDI port of the slave.
2. Tie the RXCHBONDMASTER of the master transceiver High.
3. Tie the RXCHBONDSSLAVE of each slave transceiver High.

When GTP transceivers are directly connected, meeting the timing constraints becomes difficult as the transceivers get further apart. The solution to this problem is to connect the transceivers in a daisy chain. Daisy chaining is performed using the RXCHBONDLEVEL[2:0] ports to allow additional pipeline stages between the master and the slave. The RXCHBONDO port of each slave is used as a pipeline stage in the RXCHBONDO path from the master. [Figure 4-48](#) and [Figure 4-49](#) show two daisy-chain examples.



UG482\_c4\_30\_111011

Figure 4-48: Channel Bonding Daisy Chain Example 1



UG482\_c4\_31\_111011

Figure 4-49: Channel Bonding Daisy Chain Example 2

To set up a daisy chain, the GTP transceivers are first connected using RXCHBONDO and RXCHBONDI to create a path from the RXCHBONDI port of each slave to the RXCHBONDO port of the master. The following steps describe how to set the RXCHANBONDLEVEL for the GTP transceivers in the chain:

1. Set the RXCHANBONDLEVEL of the master to 7.
2. Set the RXCHANBONDLEVEL of each slave to the RXCHANBONDLEVEL of the GTP transceiver driving the slave's RXCHBONDI port minus 1.

3. Find the slave with the lowest level. Subtract this level from the RXCHANBONDLEVEL of all GTP transceivers so that the lowest slave has level 0 and the master has the minimum level required to service all the slaves.

When the connections between channel bonding ports among GTP transceivers are being decided, the designer must remember that RXCHBONDI and RXCHBONDO belong to the RXUSRCLK clock domain. Meeting the timing constraint of RXUSRCLK becomes increasingly difficult as RXUSRCLK increases in frequency and as directly connected transceivers get further apart.

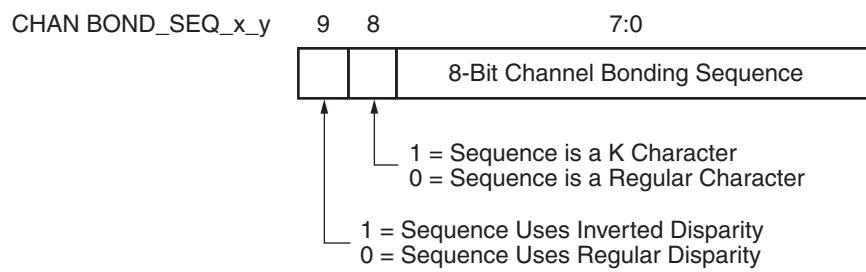
GTP transceivers in the same half of the device can be bonded with each other. A GTP transceiver located in the top half of the device can be bonded with other GTP transceivers located transceivers in the top half. A GTP transceiver located in the bottom half of the device cannot be bonded with a GTP transceiver located in the top half of the device.

As long as timing constraints are met, there is no limit to the number of GTP transceivers that can be on a particular RXCHANBONDLEVEL.

## Setting Channel Bonding Sequences

The channel bonding sequence is programmed in the same way as the clock correction sequence. CHAN\_BOND\_SEQ\_LEN sets the length of the sequence, and CHAN\_BOND\_SEQ\_1\_\* sets the values of the sequence. If CHAN\_BOND\_SEQ\_2\_USE is TRUE, CHAN\_BOND\_SEQ\_2\_\* sets the values for the alternate second sequence. The number of active bits in each subsequence depends on RX\_DATA\_WIDTH and CBCC\_DATA\_SOURCE\_SEL (see [RX Clock Correction, page 194](#)). When RX\_DISPERR\_SEQ\_MATCH is set to FALSE, CHAN\_BOND\_SEQ\_x\_y[9] is not used for matching.

[Figure 4-50](#) shows how the subsequence bits are mapped.



*Figure 4-50: Channel Bonding Sequence Settings*

As with clock correction sequences, channel bonding sequences can have don't care subsequences. CHAN\_BOND\_SEQ\_1\_ENABLE and CHAN\_BOND\_SEQ\_2\_ENABLE set these bytes. [Figure 4-51](#) shows the mapping of the enable attributes for the channel bonding subsequences.

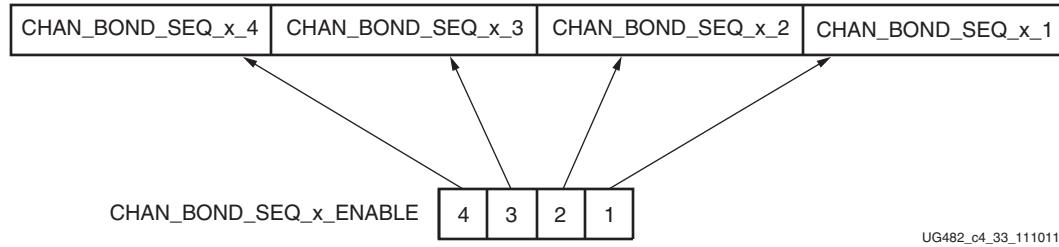


Figure 4-51: Channel Bonding Sequence Mapping

### Setting the Maximum Skew

When the master receives a channel bonding sequence, it does not trigger channel bonding immediately. Several more bytes must arrive if the slaves have more latency. This wait time effectively becomes the maximum skew that the RX elastic buffer can handle. If the skew is greater than this wait time, the slaves might not receive the sequence by the time the master triggers channel bonding.

Figure 4-52 shows two FIFOs, one for the master and one for the slave. If the slave is behind the master, the master must wait several cycles before triggering channel bonding, otherwise the slow slave does not have the channel bonding sequence in its buffer.

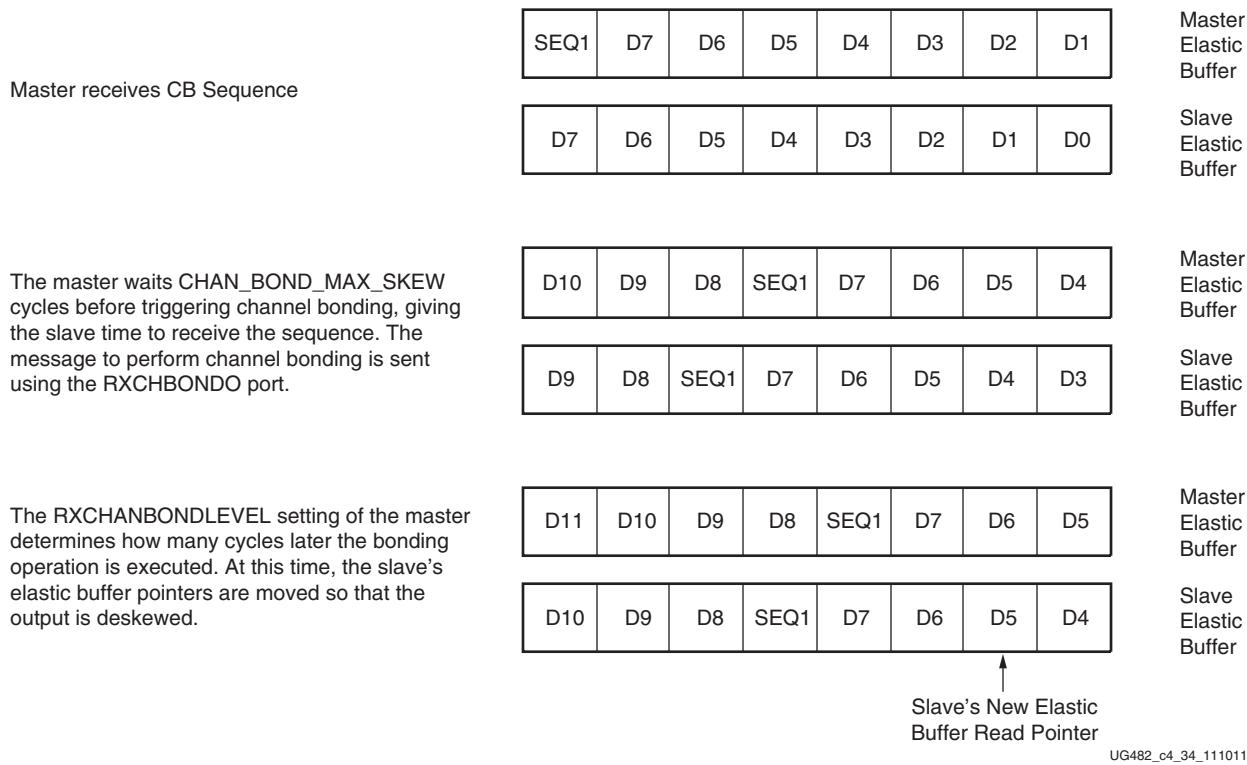


Figure 4-52: Channel Bonding Example (CHAN\_BOND\_MAX\_SKEW = 2 and Master RXCHANBONDLEVEL[2:0] = 1)

CHAN\_BOND\_MAX\_SKEW is used to set the maximum skew allowed for channel bonding sequences 1 and 2. The maximum skew range is 1 to 14. This range must always

be less than one-half the minimum distance (in bytes or 10-bit codes) between channel bonding sequences. This minimum distance is determined by the protocol being used.

## Precedence between Channel Bonding and Clock Correction

The clock correction (see [RX Clock Correction, page 194](#)) and channel bonding circuits both perform operations on the pointers of the RX elastic buffer. Normally, the two circuits work together without conflict, except when clock correction events and channel bonding events occur simultaneously. In this case, one of the two circuits must take precedence. To make clock correction a higher priority than channel bonding, CLK\_COR\_PRECEDENCE must be set to TRUE. To make channel bonding a higher priority, CLK\_COR\_PRECEDENCE must be set to FALSE.

# RX Gearbox

## Functional Description

The RX gearbox provides support for 64B/66B and 64B/67B header and payload separation. The gearbox uses output ports RXDATA[31:0] and RXHEADER[2:0] for the payload and header of the received data. Similar to [TX Gearbox, page 88](#), the RX gearbox operates with the PMA using a single clock. Because of this, occasionally, the output data is invalid. Output ports RXHEADERVALID and RXDATAVALID determine if the appropriate header and data are valid. The RX gearbox supports 2-byte and 4-byte interfaces.

The data out of the RX gearbox is not necessarily aligned. Alignment is done in the FPGA logic. The RXGEARBOXSLIP port can be used to slip the data from the gearbox cycle-by-cycle until correct alignment is reached. It takes a specific number of cycles before the bitslip operation is processed and the output data is stable. Descrambling of the data and block synchronization is done in the FPGA logic.

## Ports and Attributes

[Table 4-41](#) defines the RX gearbox ports.

*Table 4-41: RX Gearbox Ports*

Port Name	Dir	Clock Domain	Description
RXDATAVALID[1:0]	Out	RXUSRCLK2	<ul style="list-style-type: none"> <li>Bit 0: Status output when Gearbox 64B/66B or 64B/67B is used, which indicates that the data appearing on RXDATA is valid. For example, during 64B/66B encoding, this signal is deasserted every 32 cycles for the 4-byte interface and every 64 cycles for the 2-byte interface.</li> <li>Bit 1: Reserved.</li> </ul>
RXGEARBOXSLIP	In	RXUSRCLK2	<p>When High, this port causes the gearbox contents to slip to the next possible alignment. This port is used to achieve alignment with the FPGA logic. Asserting this port for one RXUSRCLK2 cycle changes the data alignment coming out of the gearbox.</p> <p>RXGEARBOXSLIP must be deasserted for at least one cycle and then reasserted to cause a new realignment of the data. If multiple realignments occur in rapid succession, it is possible to pass the proper alignment point without recognizing the correct alignment point in the FPGA logic.</p>

Table 4-41: RX Gearbox Ports (Cont'd)

Port Name	Dir	Clock Domain	Description
RXHEADER[2:0]	Out	RXUSRCLK2	Header outputs for 64B/66B (1:0) and 64B/67B (2:0).
RXHEADERVALID	Out	RXUSRCLK2	Indicates that the RXHEADER is valid when using the gearbox.
RXSTARTOFSEQ	Out	RXUSRCLK2	When the gearbox 64B/66B or 64B/67B is enabled, this output indicates when the sequence counter is 0 for the present RXDATA outputs.

Table 4-42 defines the RX gearbox attributes.

Table 4-42: RX Gearbox Attributes

Attribute	Type	Description
GEARBOX_MODE	3-bit Binary	This attribute indicates the TX and RX gearbox modes: <ul style="list-style-type: none"> <li>Bit 2: Set to 0. Unused.</li> <li>Bit 1: Set to 0.</li> <li>0: Use the external sequence counter and apply inputs to TXSEQUENCE in the TX gearbox.</li> <li>Bit 0:               <ul style="list-style-type: none"> <li>0: 64B/67B gearbox mode for Interlaken</li> <li>1: 64B/66B gearbox</li> </ul> </li> </ul>
RXGEARBOX_EN	String	When TRUE, this attribute enables the RX gearbox.

## Enabling the RX Gearbox

To enable the RX gearbox for the GTP transceiver, set the attribute RXGEARBOX\_EN to TRUE. The GEARBOX\_MODE attribute controls the GTP transceiver's TX and RX gearbox use modes.

## RX Gearbox Operating Modes

The RX gearbox supports 2-byte and 4-byte logic interfaces to the FPGA logic.

As shown in Figure 4-53, either mode uses the RXDATA, RXHEADER, RXDATAOUTVALID, and RXHEADEROUTVALID outputs in addition to the RXGEARBOXSLIP input.

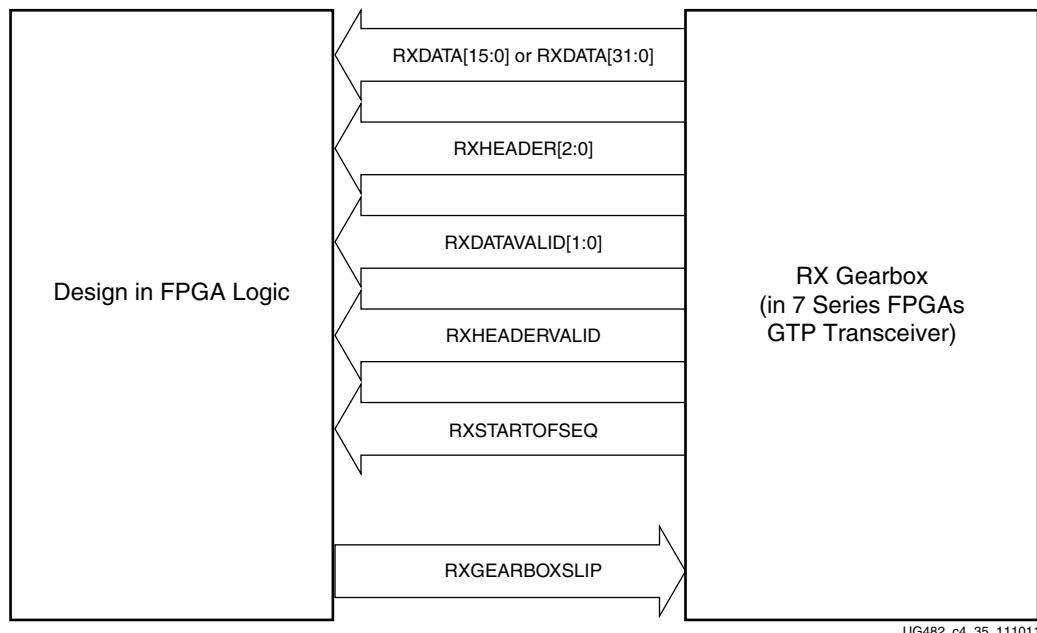


Figure 4-53: Gearbox in Either Internal or External Sequence Mode

Figure 4-54 shows an example of five cycles of data entering and exiting the RX gearbox for 64B/66B encoding when using a 2-byte logic interface (RX\_DATA\_WIDTH = 16 (2-byte)).

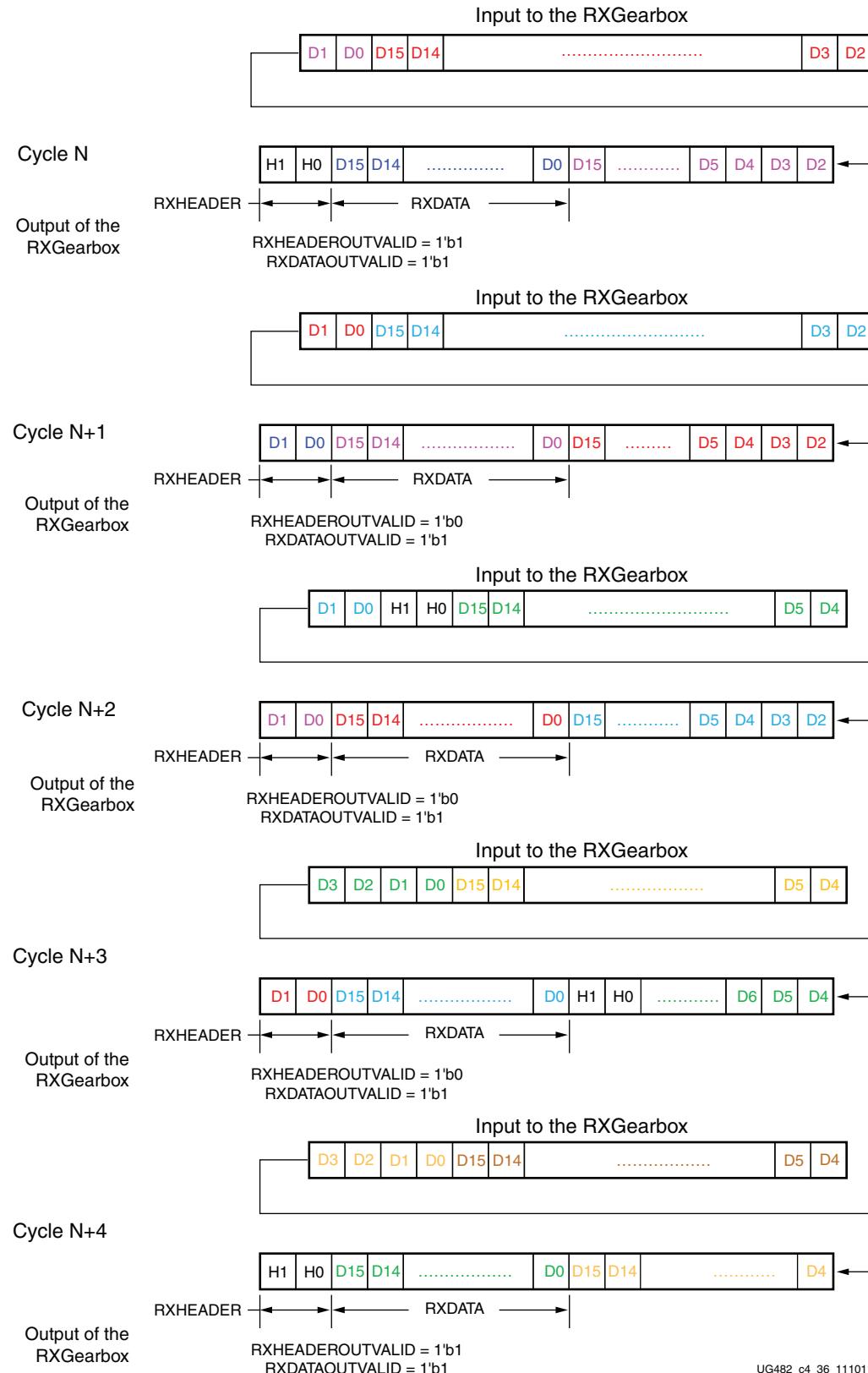
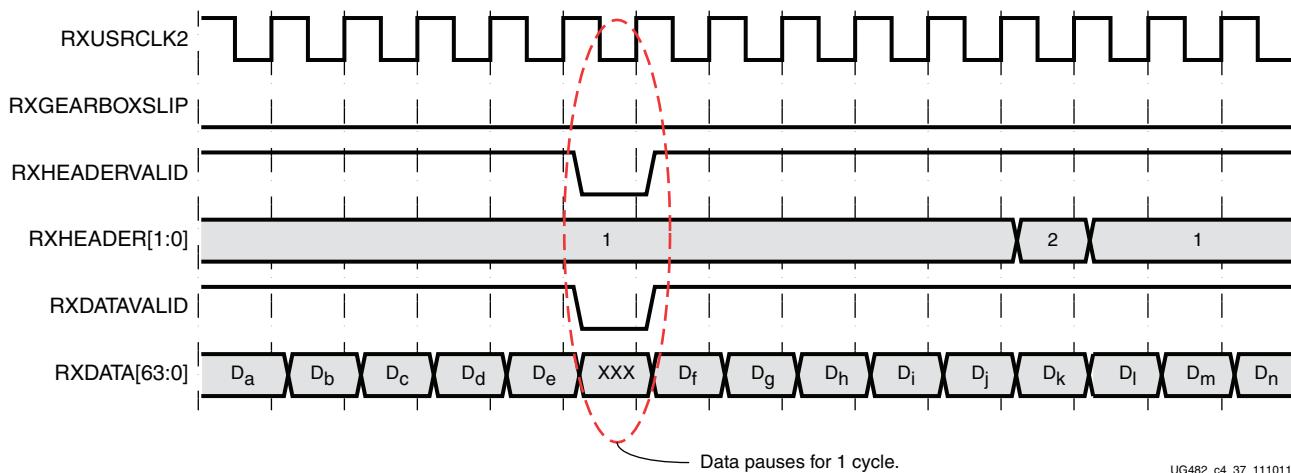


Figure 4-54: RX Gearbox Operation

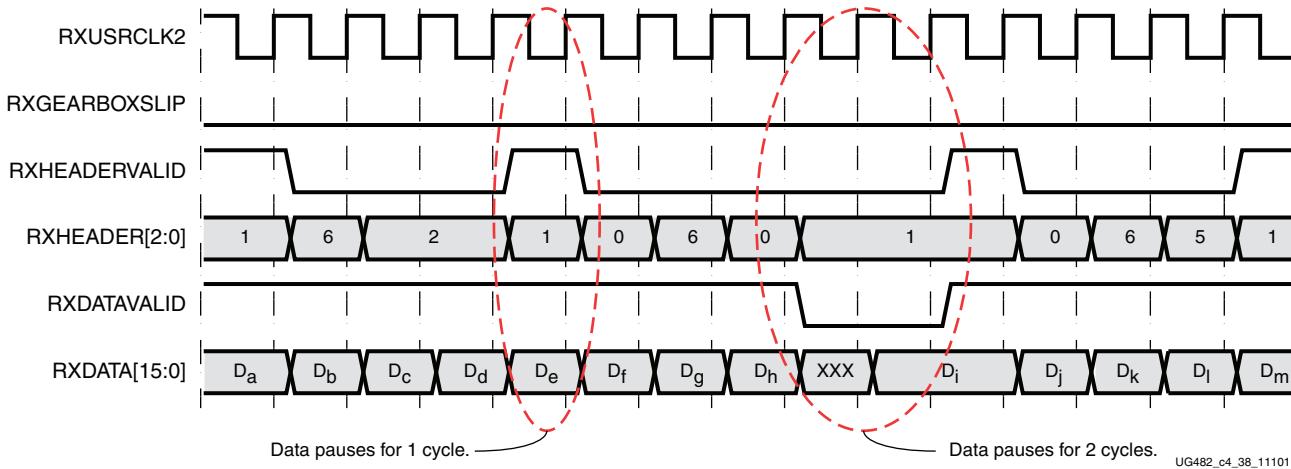
Note relevant to [Figure 4-54](#):

- As per IEEE Std 802.3ae-2002 nomenclature, H1 corresponds to RxB<0>, H0 to RxB<1>, etc.

The RX gearbox internally manages all sequencing, which differs from the TX gearbox option of either internal or external sequencing. Depending on whether a 2-byte or 4-byte interface is used, RXDATAOUTVALID and RXHEADEROUTVALID assert and deassert for different periods of length. The RX gearbox encounters similar data and header pauses found in the TX gearbox. [Figure 4-55](#) shows such a pause in addition to RXHEADERVALID and RXDATAVALID being deasserted for one cycle. [Figure 4-56](#) shows the operation for 64B/67B encoding when RX\_DATA\_WIDTH = 16 (2-byte).



[Figure 4-55: RX Gearbox When Using 64B/66B Encoding and RX\\_DATA\\_WIDTH = 32 \(4-Byte\)](#)

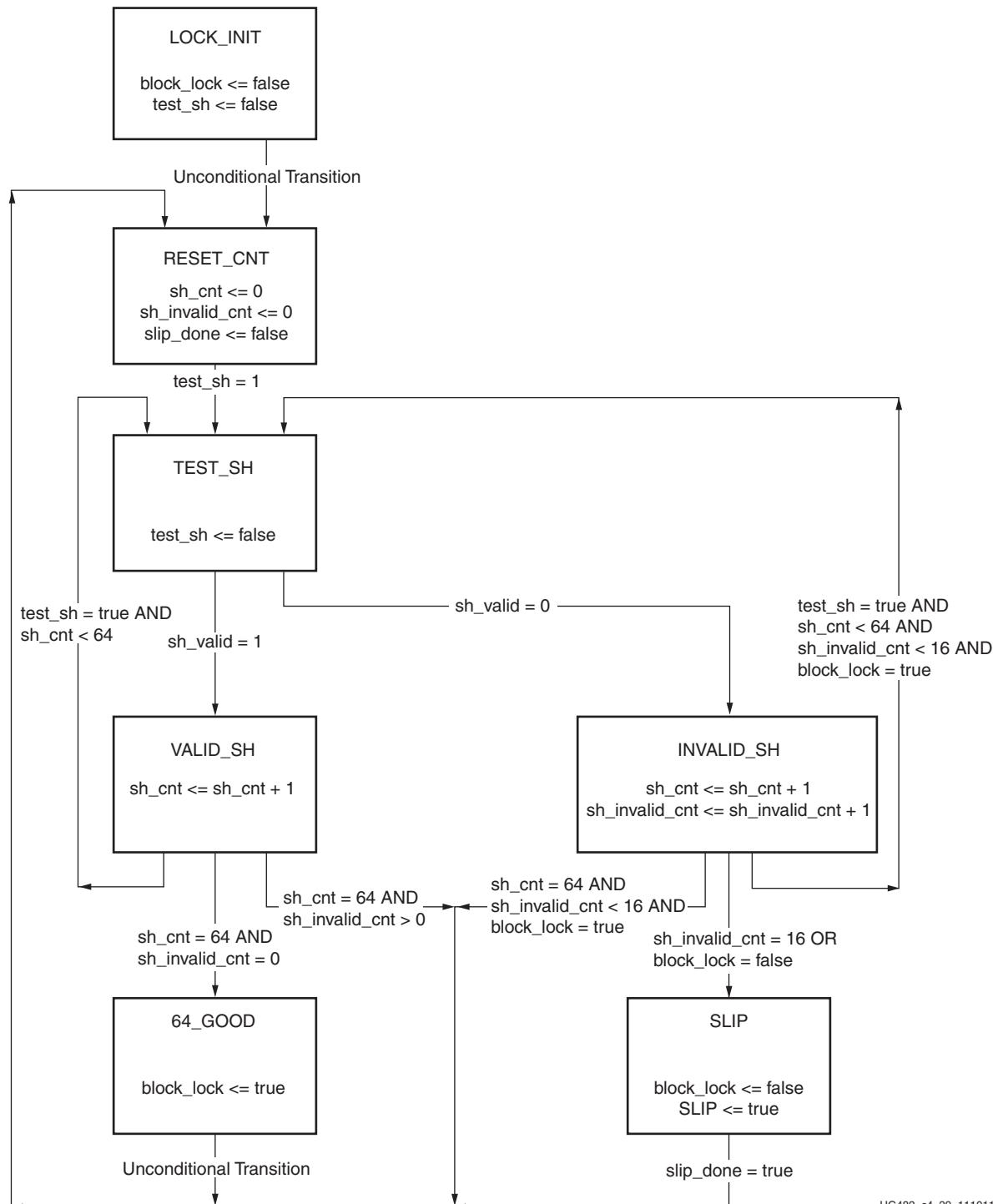


[Figure 4-56: RX Gearbox When Using 64B/67B Encoding and RX\\_DATA\\_WIDTH = 16 \(2-Byte\)](#)

## RX Gearbox Block Synchronization

The 64B/66B and 64B/67B protocols depend on block synchronization to determine their block boundaries. Block synchronization is required because all incoming data is unaligned before block lock is achieved. The goal is to search for the valid synchronization header by changing the data alignment. The RXGEARBOXSLIP input port is used to

change the gearbox data alignment so that all possible alignments can be checked. The RXGEARBOXSLIP signal feeds back from the block synchronization state machine to the RX gearbox and tells it to slip the data alignment. This process of slipping and testing the synchronization header repeats until block lock is achieved. When using the RX gearbox, a block synchronization state machine is required in the FPGA logic. [Figure 4-57](#) shows the operation of a block synchronization state machine. The 7 Series FPGAs Transceivers Wizard has example code for this type of module.



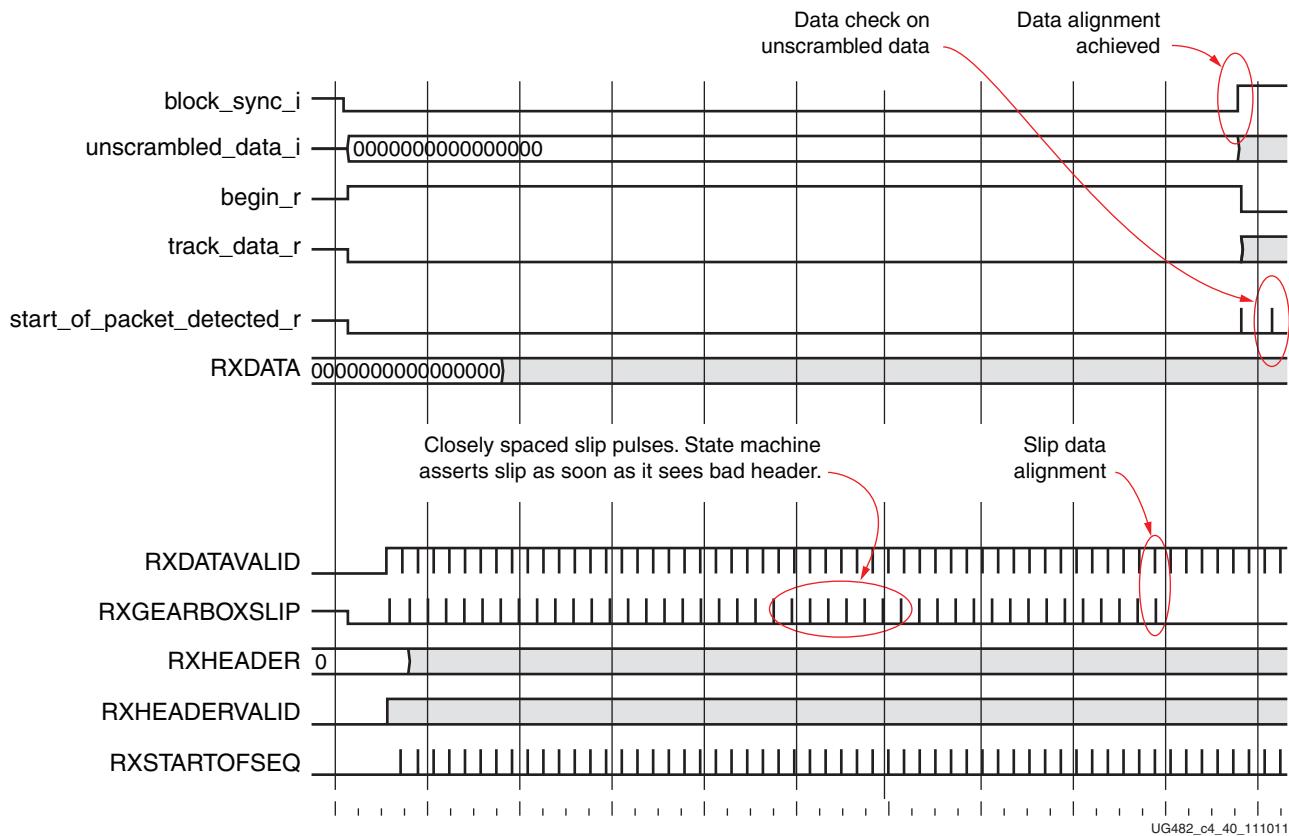
UG482\_c4\_39\_111011

**Figure 4-57: Block Synchronization State Machine**

The state machine works by keeping track of valid and invalid synchronization headers. Upon reset, block lock is deasserted, and the state is **LOCK\_INIT**. The next state is **RESET\_CNT** where all counters are zeroed out. The synchronization header is analyzed in the **TEST\_SH** state. If the header is valid, `sh_cnt` is incremented in the **VALID\_SH** state, otherwise `sh_count` and `sh_invalid_count` are incremented in the **INVALID\_SH** state.

For the block synchronization state machine shown in [Figure 4-57](#), sh\_cnt\_max and sh\_invalid\_cnt\_max are both constants that are set to 64 and 16, respectively. From the VALID\_SH state, if sh\_cnt is less than the value sh\_cnt\_max and test\_sh is High, the next state is TEST\_SH. If sh\_cnt is equal to sh\_cnt\_max and sh\_invalid\_cnt equals 0, the next state is GOOD\_64 and from there block\_lock is asserted. Then the process repeats again and the counters are cleared to zeros. To achieve block lock, the state machine must receive sh\_cnt\_max number of valid synchronization headers in a row without getting an invalid synchronization header. However, when block lock is achieved sh\_invalid\_cnt\_max – 1, the number of invalid synchronization headers can be received within sh\_cnt\_max number of valid synchronization headers. Thus, once locked, it is harder to break lock.

[Figure 4-58](#) shows a waveform of the block synchronization state machine asserting RXGEARBOXSLIP numerous times because of invalid synchronization headers before achieving data alignment. After the RXGEARBOXSLIP is issued, the state machine waits 32 RXUSRCLK2 cycles before checking for valid synchronization headers.



[Figure 4-58: RX Gearbox with Block Synchronization](#)

## FPGA RX Interface

### Functional Description

The FPGA RX interface is the FPGA's gateway to the RX datapath of the GTP transceiver. Applications receive data through the GTP transceiver by reading data from the RXDATA port on the positive edge of RXUSRCLK2. The width of the port can be configured to be two or four bytes wide. The actual width of the port depends on the RX\_DATA\_WIDTH

attribute and RX8B10BEN port setting. Port widths can be 16, 20, 32, and 40 bits. The rate of the parallel clock (RXUSRCLK2) at the interface is determined by the RX line rate, the width of the RXDATA port, and whether or not 8B/10B decoding is enabled. In some operating modes, a second parallel clock (RXUSRCLK) must be provided for the internal PCS logic in the transmitter. This section shows how to drive the parallel clocks and explains the constraints on those clocks for correct operation.

## Interface Width Configuration

The 7 series GTP transceiver contains a 2-byte internal datapath. The FPGA interface width is configurable by setting the RX\_DATA\_WIDTH attribute. When the 8B/10B decoder is enabled, RX\_DATA\_WIDTH must be configured to 20 bits or 40 bits, and in this case, the FPGA RX interface only uses the RXDATA port. For example, RXDATA[15:0] is used when the FPGA interface width is 16. When the 8B/10B decoder is bypassed, RX\_DATA\_WIDTH can be configured to any of the available widths: 16, 20, 32, or 40 bits.

**Table 4-43** shows how the interface width for the RX datapath is selected. 8B/10B decoding is described in more detail in [RX 8B/10B Decoder, page 173](#).

**Table 4-43: FPGA RX Interface Datapath Configuration**

RX8B10BEN	RX_DATA_WIDTH	FPGA Interface Width	Internal Data Width
1	20	16	20
	40	32	20
0	16	16	16
	20	20	20
	32	32	16
	40	40	20

When the 8B/10B decoder is bypassed and RX\_DATA\_WIDTH is 20 or 40, the RXDISPERR and RXCHARISK ports are used to extend the RXDATA port from 16 to 20 bits, or 32 to 40 bits. **Table 4-44** shows the data received when the 8B/10B decoder is disabled. When the RX gearbox is used, refer to [RX Gearbox, page 211](#) for data transmission order.

**Table 4-44: RX Data Received When the 8B/10B Decoder is Bypassed**

Data Received	<<< Data Reception is Right to Left (LSB to MSB) >>>																																								
	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RXDISPERR[3]	RXDISPERR[3]	RXDATA[31:24]										RXDISPERR[2]	RXCHARISK[2]	RXDATA[23:16]										RXDISPERR[1]	RXCHARISK[1]	RXDATA[15:8]										RXDISPERR[0]	RXCHARISK[0]	RXDATA[7:0]			

## RXUSRCLK and RXUSRCLK2 Generation

The FPGA RX interface includes two parallel clocks: RXUSRCLK and RXUSRCLK2. RXUSRCLK is the internal clock for the PCS logic in the GTP transceiver transmitter. The required rate for RXUSRCLK depends on the internal datapath width of the GTPE2\_CHANNEL primitive and the RX line rate of the GTP transceiver transmitter. [Equation 4-2](#) shows how to calculate the required rate for RXUSRCLK.

$$\text{RXUSRCLK Rate} = \frac{\text{Line Rate}}{\text{Internal Datapath Width}}$$
Equation 4-2

RXUSRCLK2 is the main synchronization clock for all signals into the RX side of the GTP transceiver. Most signals into the RX side of the GTP transceiver are sampled on the positive edge of RXUSRCLK2. RXUSRCLK2 and RXUSRCLK have a fixed-rate relationship based on the RX\_DATA\_WIDTH setting. [Table 4-45](#) shows the relationship between RXUSRCLK2 and RXUSRCLK per RX\_DATA\_WIDTH value.

**Table 4-45: RXUSRCLK2 Frequency Relationship to RXUSRCLK**

FPGA Interface Width	RX_DATA_WIDTH	RXUSRCLK2 Frequency
2-Byte	16, 20	$F_{\text{RXUSRCLK2}} = F_{\text{RXUSRCLK}}$
4-Byte	32, 40	$F_{\text{RXUSRCLK2}} = F_{\text{RXUSRCLK}} / 2$

These rules about the relationships between clocks must be observed for RXUSRCLK and RXUSRCLK2:

- RXUSRCLK and RXUSRCLK2 must be positive-edge aligned, with as little skew as possible between them. As a result, low-skew clock resources (BUFGs and BUFHs) should be used to drive RXUSRCLK and RXUSRCLK2.
- If the channel is configured so the same oscillator drives the reference clock for the transmitter and the receiver, TXOUTCLK can be used to drive RXUSRCLK and RXUSRCLK2 in the same way that they are used to drive TXUSRCLK and TXUSRCLK2. When clock correction is turned off or the RX buffer is bypassed, RX phase alignment must be used to align the serial clock and the parallel clocks.
- If separate oscillators are driving the reference clocks for the transmitter and receiver on the channel, and clock correction is not used, RXUSRCLK and RXUSRCLK2 must be driven by RXOUTCLK (RXOUTCLKSEL = 3'b010 for RXOUTCLKPMA), and the phase-alignment circuit must be used.
- If clock correction is used, RXUSRCLK and RXUSRCLK2 can be sourced by RXOUTCLK or TXOUTCLK.

## Ports and Attributes

[Table 4-46](#) defines the FPGA RX interface ports.

**Table 4-46: FPGA RX Interface Ports**

Port	Dir	Clock Domain	Description
RXDISPERR[3:0]	Out	RXUSRCLK2	When 8B/10B decoding is disabled, RXDISPERR is used to extend the data bus for 20-bit and 40-bit RX interfaces.
RXCHARISK[3:0]	Out	RXUSRCLK2	When 8B/10B decoding is disabled, RXCHARISK is used to extend the data bus for 20-bit and 40-bit RX interfaces.

Table 4-46: FPGA RX Interface Ports (Cont'd)

Port	Dir	Clock Domain	Description
RXDATA[31:0]	Out	RXUSRCLK2	<p>The bus for transmitting data. The width of this port depends on RX_DATA_WIDTH:</p> <ul style="list-style-type: none"> <li>RX_DATA_WIDTH = 16, 20: RXDATA[15:0] = 16 bits wide</li> <li>RX_DATA_WIDTH = 32, 40: RXDATA[31:0] = 32 bits wide</li> </ul> <p>When a 20-bit or 40-bit bus is required, the RXCHARISK and RXDISPERR ports from the 8B/10B encoder are concatenated with the RXDATA port. See <a href="#">Table 4-44, page 219</a>.</p>
RXUSRCLK	In	Clock	This port is used to provide a clock for the internal RX PCS datapath.
RXUSRCLK2	In	Clock	This port is used to synchronize the FPGA logic with the RX interface. This clock must be positive-edge aligned to RXUSRCLK when RXUSRCLK is provided by the user.

[Table 4-47](#) defines the FPGA RX interface attributes.

Table 4-47: FPGA RX Interface Attributes

Attribute	Type	Description
RX_DATA_WIDTH	Integer	<p>Sets the bit width of the RXDATA port. When 8B/10B encoding is enabled, RX_DATA_WIDTH must be set to 20 or 40. Valid settings are 16, 20, 32, or 40.</p> <p>See <a href="#">Interface Width Configuration, page 219</a> for more details.</p>



# Board Design Guidelines

---

## Overview

Topics related to implementing a design on a printed circuit board using the 7 series Artix™-7 FPGA GTP transceivers are presented in this chapter. The GTP transceivers are analog circuits that require special consideration and attention when designing and implementing them on a printed circuit board. Besides an understanding of the functionality of the device pins, a design that performs optimally requires attention to issues such as device interfacing, transmission line impedance and routing, power supply design filtering and distribution, component selection, and PCB layout and stackup design.

## Pin Description and Design Guidelines

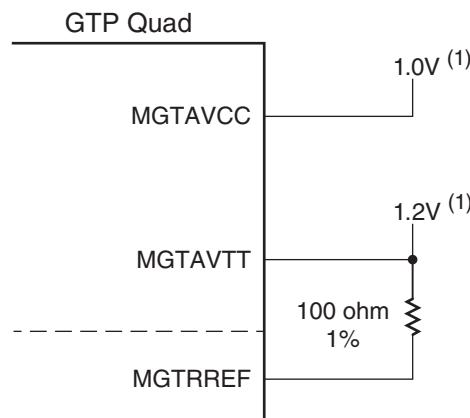
### GTP Pin Descriptions

*Table 5-1: GTP Quad Pin Descriptions*

Pins	Direction	Description
MGTREFCLK0P MGTREFCLK0	In (Pad)	Differential clock input pin pair for the reference clock of the GTP transceiver Quad.
MGTREFCLK1P MGTREFCLK1N	In (Pad)	Differential clock input pin pair for the reference clock of the GTP transceiver Quad.
MGTPRXP0/MGTPRXN0 MGTPRXP1/MGTPRXN1 MGTPRXP2/MGTPRXN2 MGTPRXP3/MGTPRXN3	In (Pad)	RXP and RXN are the differential input pairs for each of the receivers in the GTP transceiver Quad.
MGTTXP0/MGTPTXN0 MGTTXP1/MGTPTXN1 MGTTXP2/MGTPTXN2 MGTTXP3/MGTPTXN3	Out (Pad)	TXP and TXN are the differential output pairs for each of the transmitters in the GTP transceiver Quad.
MGTRREF	In (Pad)	Calibration resistor input pin for the termination resistor calibration circuit. Connect to a $100\Omega$ resistor that is also connected to MGTAVTT.

Table 5-1: GTP Quad Pin Descriptions (Cont'd)

Pins	Direction	Description
MGTAVCC	In (Pad)	MGTAVCC is the analog supply for the internal analog circuits of the GTP Quad tile. This includes the analog circuits for the PLLs, transmitters and receivers. Most packages have multiple groups of power supply connections in the package for MGTAVCC. Refer to the package pin definitions to identify in which power supply group a specific GTP Quad is assigned. Nominal voltage = 1.0 VDC.
MGTAVTT	In (Pad)	MGTAVTT is the analog supply for the transmitter and receiver termination circuits of the GTP Quad tile. Most packages have multiple groups of power supply connections in the package for MGTAVTT. Refer to the package pin definitions to identify which power supply group a specific GTP Quad is assigned. Nominal voltage = 1.2 VDC.

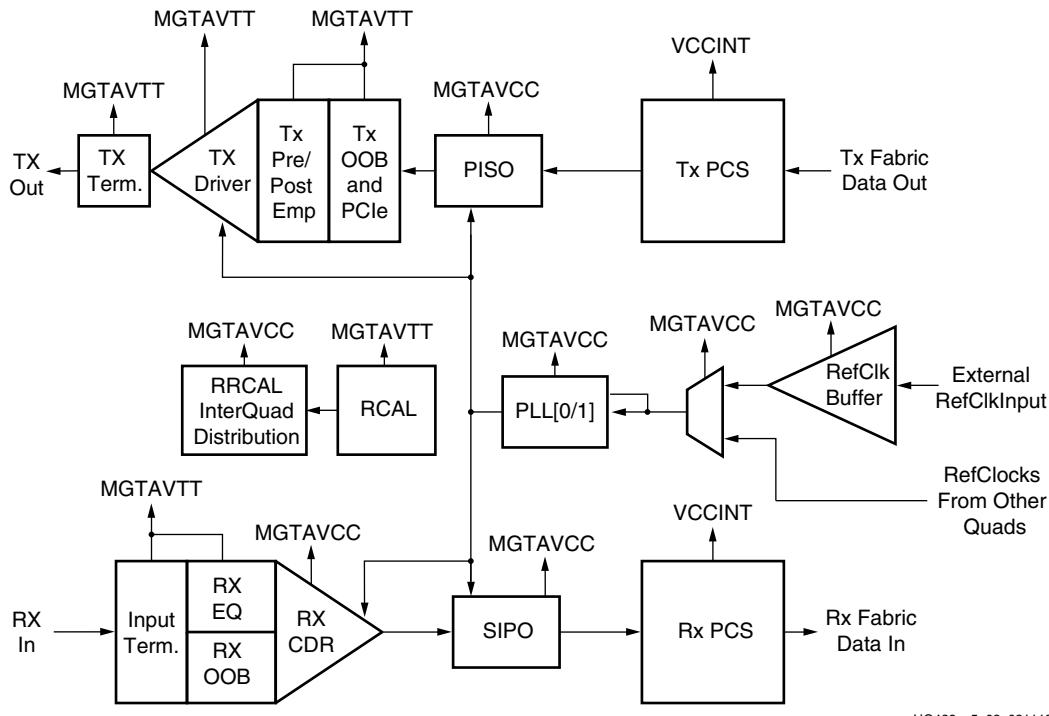


UG482\_c5\_01\_080612

Figure 5-1: Artix-7 FPGA GTP Power Supply Connections

Notes relative to Figure 5-1:

1. Nominal values. Refer to [DS181](#), Artix-7 FPGAs Data Sheet: DC and Switching Characteristics for values and tolerances.



UG482\_c5\_02\_021113

Figure 5-2: 7 Series FPGAs GTP Transceiver Internal Power Supply Connections

## Termination Resistor Calibration Circuit

There is one resistor calibration circuit (RCAL) for each GTP Quad. The MGTRREF pin is used to connect the bias circuit power and the external calibration resistor to the RCAL circuit. The RCAL circuit performs the resistor calibration only during configuration of the FPGA. Prior to configuration all analog supply voltages must be present and within the proper tolerance as specified in [DS181, Artix-7 FPGAs Data Sheet: DC and Switching Characteristics](#).

The MGTRREF pin should be connected to the MGTAVTT supply through a  $100\Omega$  precision external resistor. The resistor calibration circuit provides a controlled current load to the resistor that is connected to the MGTRREF pin. It then senses the voltage drop across the external calibration resistor and uses that value to adjust the internal resistor calibration setting. The quality of the resistor calibration is dependent on the accuracy of the voltage measurement at the MGTRREF pin.

## Managing Unused GTP Transceivers

In many applications only a portion of the GTP Transceivers are required. There are considerations for managing the unused GTP Transceivers that affect such things as power consumption of the Artix-7 FPGA. When considering which GTP Quads to use in an application, the organization of the package power planes should be taken into account. Having multiple analog power planes in the package allows for efficient utilization of power. If only a small portion of the GTP Quads will be used then it might be possible to leave some of the GTP Quads completely un-powered.

## Analog Power Supply Pins

The Artix-7 FPGA GTP Quad analog power supplies, MGTAVCC and MGTAVTT, have planes inside the package. For some of the packages, there are multiple planes for each of these analog power supplies. These planes are organized into groups with group designators such as G10 and G11. For each of the GTP Quads in the power supply group, the power supply pins for each GTP quad are connected to the same planes inside the package. For each power supply group, there is an MGTAVCC power plane in the package and there is an MGTAVTT power supply plane in the package. (See [Table 5-2](#) for a description of the connected Artix-7 FPGA GTP Quads.)

Notice in [Table 5-2](#) that there are two types of groupings of the GTP Quads in a column.

- Devices with two power supply groups for GTP Quads in the package
  - G10 power supply group for a set of GTP Quads
  - G11 power supply group for a different set of GTP Quads
- Devices with a single power supply group for a GTP Quad

**Table 5-2: Artix-7 Devices, Packages, GTP Transceivers, and Power Planes**

	Artix-7 FPGA GTP Transceivers			
	MGT113	MGT213	MGT116	MGT216
XC7A15T-CPG326				Single <sup>(1)</sup>
XC7A15T-CSG325				Single
XC7A15T-FGG484				Single
XC7A35T-CPG236				Single
XC7A35T-CSG325				Single
XC7A35T-FGG484				Single
XC7A50T-CPG236				Single
XC7A50T-CSG325				Single
XC7A50T-FGG484				Single
XC7A75T-FGG484				Single
XC7A75T-FGG676		G10 <sup>(2)</sup>		G11 <sup>(3)</sup>
XC7A100T-FGG484				Single
XC7A100T-FGG676		G10		G11
XC7A200T-SBG484				Single
XC7A200T-FBG484				Single
XC7A200T-FBG676		G10		G11

**Table 5-2: Artix-7 Devices, Packages, GTP Transceivers, and Power Planes**

	Artix-7 FPGA GTP Transceivers			
	MGT113	MGT213	MGT116	MGT216
XC7A200T-FFG1156	G10	G10	G11	G11

**Notes:**

1. Single: GTP Quad is powered by a single set of power planes (MGTAVCC, MGTAVTT).
2. G10: GTP Quad is powered by the G10 set of package power planes. (MGTAVCC\_G10, MGTAVTT\_G10).
3. G11: GTP Quad is powered by the G11 set of package power planes. (MGTAVCC\_G11, MGTA References to the XC part numbers also apply to the XA and XQ part numbers where available. VTT\_G11).
4. References to the XC part numbers also apply to the XA and XQ part numbers where available.

Zynq®-7000 AP SoC connected GTP Quads are described in [Table 5-3](#).

**Table 5-3: Zynq-7000 Devices, Packages, GTP Transceivers, and Power Planes**

	Zynq-7000 AP SoC GTP Transceivers	
	MGT112	
XC7Z015-CLG485	Single	

[Figure 5-3](#) shows the orientation in the device package pin field of the GTP Quads and power supply groups. The grouping for the GTP Quads depends on the device package. The FGG484, FBG484, CPG236, CLG484, CSG325, and SBG484 packages have a single common power plane for each of the MGT supplies. The FGG676 and FBG676 packages have two power planes for each of the MGT supplies. The FFG1156 package also has two power planes for each of the MGT supplies.

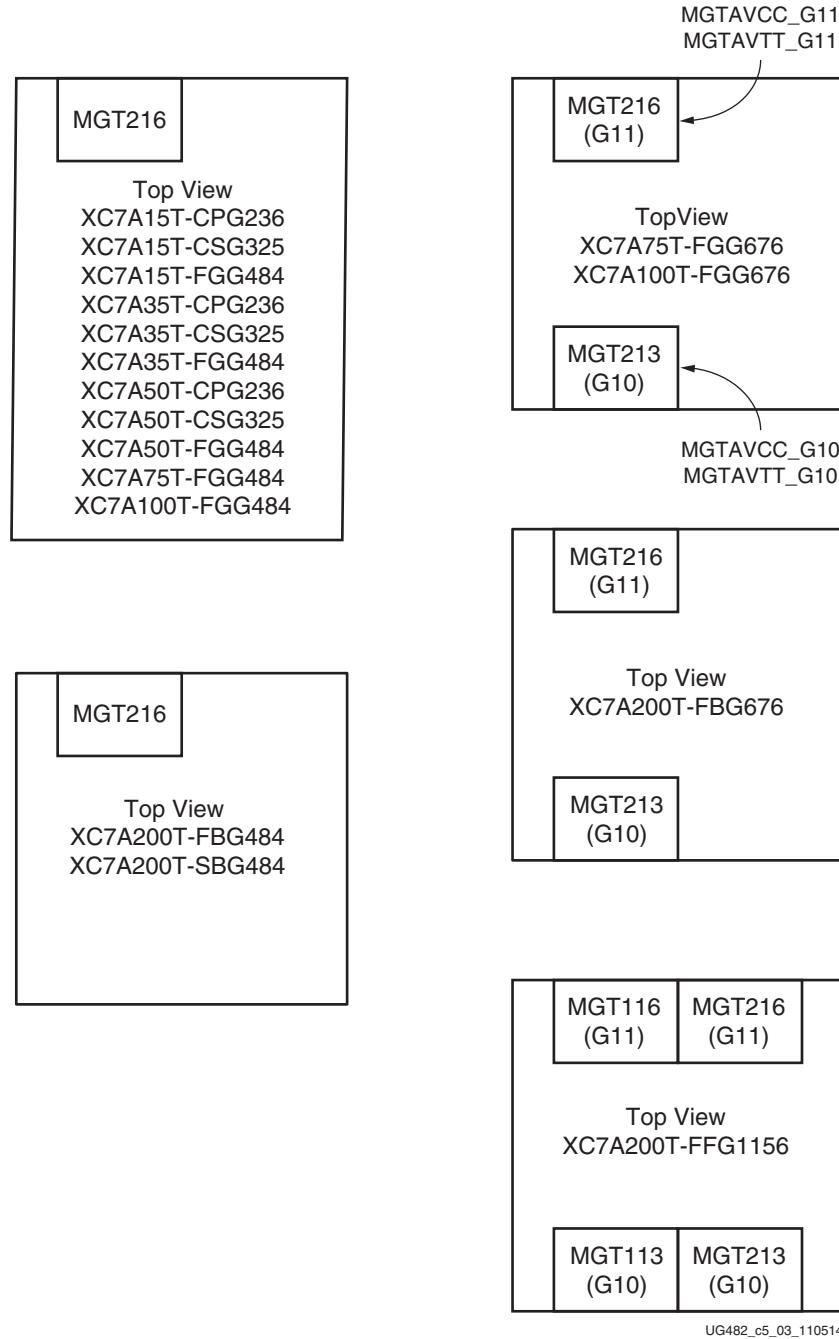


Figure 5-3: GTP Quad and Package Power Plane Group Orientation in Artix-7 FPGA Packages

Notes relative to Figure 5-3:

1. (G10) and (G11) denote package power planes. The GTP Quad power connections are common to all GTP Quads that have the same package power plane. Devices without this notation have only one set of GTP package power planes.

## Unused GTP Quad Power Supply Group

If none of the GTP Quads in a power supply group are used in the application, the GTP Quad device pins can be connected as shown in [Table 5-4](#).

**Table 5-4: Unused GTP Quad Column Connections**

Pin or Pin Pair of the Unused GTP Quad	Connection
MGTAVCC	GND
MGTAVTT	GND
MGTREFCLKP/MGTREFCLKN	Float
MGTRXP/MGTRXN	GND
MGTTXP/MGTTXN	Float
MGTRREF <sup>(1)</sup>	GND

**Notes:**

1. This is the only scenario when the MGTRREF pins can be connected to ground. In all other scenarios, these pins must be connected for normal operation.

## Partially Unused GTP Quad Power Supply Group

If only a portion of the GTP Quads in a power supply group is used, the pins for unused GTP Quads must be connected as shown in [Table 5-5](#). Notice for this case, the power supply pins must be connected to the appropriate power supply operating voltage and the MGTRREF pin is connected to MGTAVTT through a  $100\Omega$  resistor.

**Table 5-5: Unused GTP Quad Column Connections**

Pin or Pin Pair of the Unused GTP Quad	Connection
MGTAVCC	AVCC
MGTAVTT	AVTT
MGTREFCLKP/MGTREFCLKN	Float
MGTRXP/MGTRXN	GND
MGTTXP/MGTTXN	Float
MGTRREF	MGTAVTT through a $100\Omega$ resistor

## Partially Used GTP Quad

There are four GTP Transceivers in an Artix-7 FPGA GTP Quad. For a partially used GTP Quad where one or more but not all of the transceivers are not used, the analog power supplies, MGTAVCC and MGTAVTT, need to be connected. [Table 5-6](#) shows the connections to the Quad for the GTP transceivers that are not used.

**Table 5-6: Unused GTP Quad Column Connections**

Pin or Pin Pair of the Unused GTP Quad	Connection
MGTAVCC	AVCC
MGTAVTT	AVTT

Table 5-6: Unused GTP Quad Column Connections (Cont'd)

Pin or Pin Pair of the Unused GTP Quad	Connection
MGTREFCLKP/MGTREFCLKN	Float (if not used)
MGTRXP/MGTRXN	GND
MGTTXP/MGTTXN	Float
MGTRREF	MGTAVTT through a 100Ω resistor

## Reference Clock

### Overview

This section focuses on the selection of the reference clock source or oscillator. An oscillator is characterized by:

- Frequency range
- Output voltage swing
- Jitter (deterministic, random, peak-to-peak)
- Rise and fall times
- Supply voltage and current
- Noise specification
- Duty cycle and duty-cycle tolerance
- Frequency stability

These characteristics are selection criteria when choosing an oscillator for a GTP transceiver design. [Figure 5-4](#) illustrates the convention for a single-ended clock input voltage swing, peak-to-peak. This figure is provided to show the contrast to the differential clock input voltage swing calculation shown in [Figure 5-5](#).

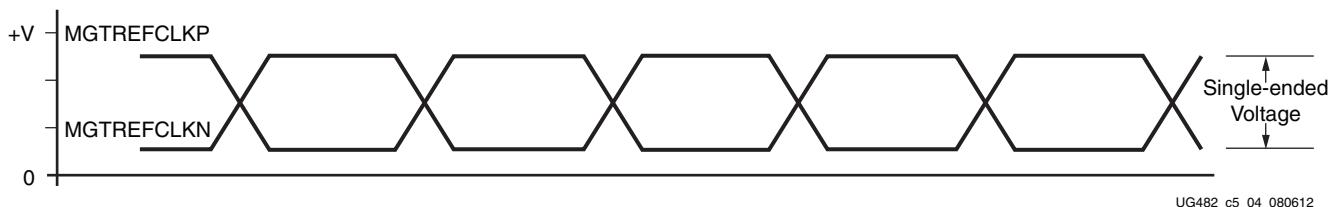
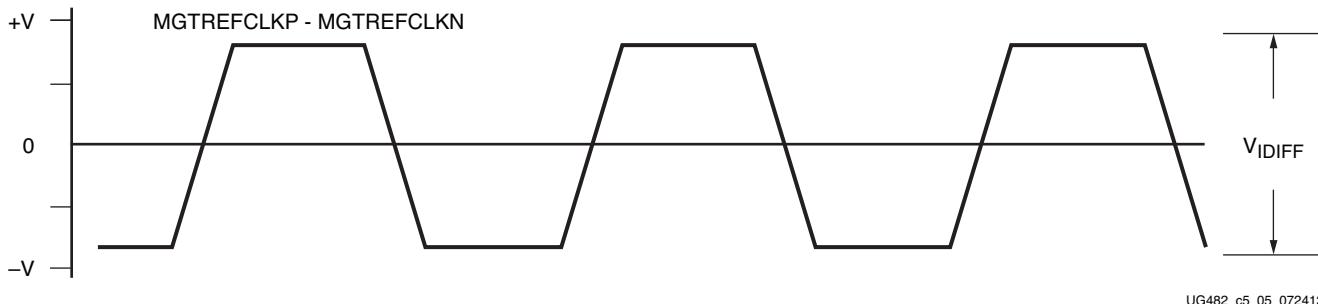


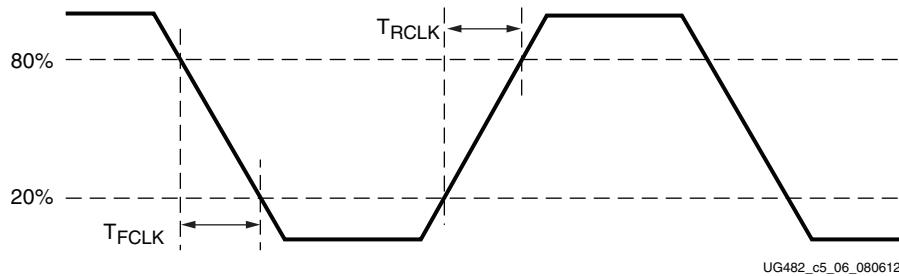
Figure 5-4: Single-Ended Clock Input Voltage Swing, Peak-to-Peak

[Figure 5-5](#) illustrates the differential clock input voltage swing, peak-to-peak, which is defined as MGTREFCLKP - MGTREFCLKN and used in the GTP transceiver portion of the Artix-7 FPGA data sheet.



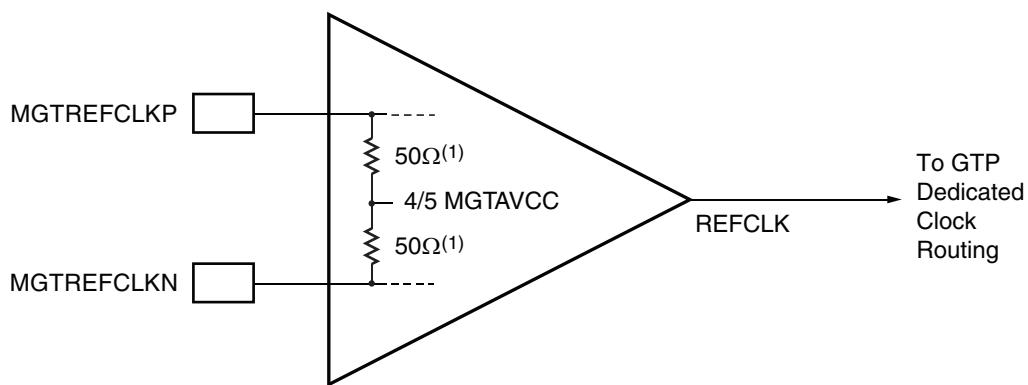
**Figure 5-5: Differential Clock Input Voltage Swing, Peak-to-Peak**

[Figure 5-6](#) shows the rise and fall time convention of the reference clock.



**Figure 5-6: Rise and Fall Time**

[Figure 5-7](#) illustrates the internal details of the IBUFDS. The dedicated differential reference clock input pair MGTREFCLKP/MGTREFCLKN is internally terminated with  $100\Omega$  differential impedance. The common mode voltage of this differential reference clock input pair is  $4/5$  of MGTAVCC, or nominal 0.8V. Refer to [DS181, Artix-7 FPGAs Data Sheet: DC and Switching Characteristics](#) for exact specifications.



UG482\_c5\_07\_080612

**Figure 5-7: MGTREFCLK Input Buffer Details**

Notes relative to [Figure 5-7](#):

1. Nominal values. Refer to [DS181](#), *Artix-7 FPGAs Data Sheet: DC and Switching Characteristics* for values and tolerances.

## GTP Reference Clock Checklist

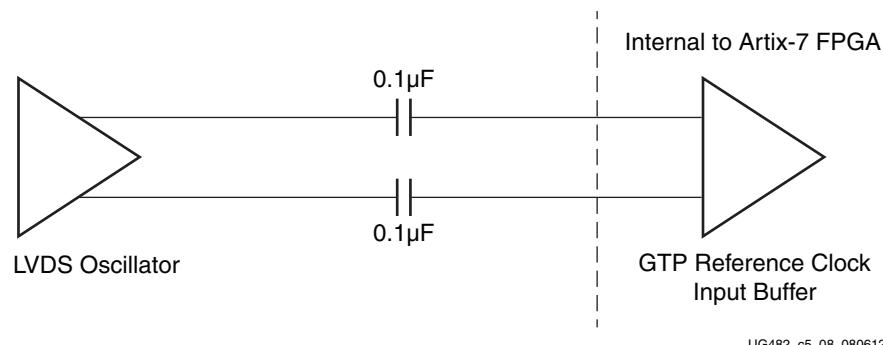
The following criteria must be met when choosing an oscillator for a design with GTP transceivers:

- Provide AC coupling between the oscillator output pins and the dedicated GTP Quad clock input pins.
- Ensure that the differential voltage swing of the reference clock is the range as specified in [DS181](#), (the nominal range is 350 mV – 2,000 mV, and the nominal typical value is 1,200 mV).
- Meet or exceed the reference clock characteristics as specified in [DS181](#).
- Meet or exceed the reference clock characteristics as specified in the standard for which the GTP transceiver provides physical layer support.
- Fulfill the oscillator vendor's requirement regarding power supply, board layout, and noise specification.
- Provide a dedicated point-to-point connection between the oscillator and GTP Quad clock input pins.
- Keep impedance discontinuities on the differential transmission lines to a minimum (impedance discontinuities generate jitter).

## Reference Clock Interface

### LVDS

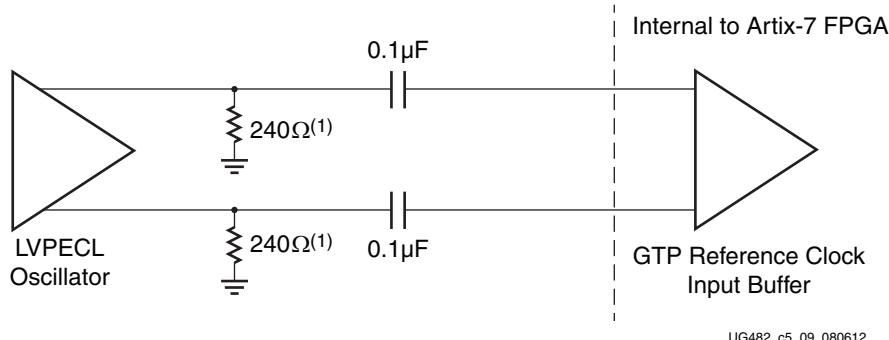
See [Figure 5-8](#).



**Figure 5-8: Interfacing LVDS Oscillator to Artix-7 FPGA GTP Reference Clock Input**

## LVPECL

See [Figure 5-9](#).



*Figure 5-9: Interfacing LVPECL Oscillator to Artix-7 FPGA GTP Reference Clock Input*

Notes relative to [Figure 5-9](#):

1. Nominal values. Refer to the oscillator vendor data sheet for actual bias resistor requirement.

## AC Coupled Reference Clock

AC coupling of the oscillator reference clock output to the GTP Quad reference clock inputs serves multiple purposes:

- Blocking a DC current between the oscillator and the GTP Quad dedicated clock input pins (which reduces the power consumption of both parts as well).
- Common mode voltage independence.
- The AC coupling capacitor forms a high-pass filter with the on-chip termination that attenuates a wander of the reference clock.

To minimize noise and power consumption, external AC coupling capacitors between the sourcing oscillator and the GTP Quad dedicated clock reference clock input pins are required.

## Unused Reference Clocks

It is recommended to leave the unused differential input pin clock pair floating (both MGTREFCLKP and MGTREFCLKN).

## Reference Clock Power

The GTP reference clock input circuit is powered by MGTAVCC. Excessive noise on this supply will have a negative impact on the performance of any GTP Quad that uses the reference clock from this circuit.

# Power Supply and Filtering

## Overview

The Artix-7 FPGA GTP Quad requires two analog power supplies: MGTAVCC at a nominal voltage level of 1.0 VDC, and MGTAVTT at a nominal voltage level of 1.2 VDC. The pins for each of these analog power supplies are tied to a plane in the package. In some packages, there are two planes for each of the analog power supplies. See [Analog Power Supply Pins, page 226](#) for a discussion of the internal power planes in the Artix-7 FPGA GTP packages.

Noise on the GTP analog power supplies can cause degradation in the performance of the transceivers. The most likely form of degradation is an increase in jitter at the output of the GTP transmitter and reduced jitter tolerance in the receiver. Sources of power supply noise are:

- Power supply regulator noise
- Power distribution network
- Coupling from other circuits

Each of these noise sources must be considered in the design and implementation of the GTP analog power supplies. The total peak to peak noise as measured at the input pin of the FPGA should not exceed 10 mV<sub>PK-PK</sub>.

## Power Supply Regulators

In many applications, the GTP analog voltage supplies have local power supply regulators that provide a final stage of voltage regulation. Preferably these regulators are placed as close as is feasible to the GTP power supply pins. Minimizing the distance between the analog voltage regulators and the GTP power supply pins reduces the opportunity for noise coupling into the supply after the regulator and for noise generated by current transients caused by load dynamics.

## Linear vs. Switching Regulators

The type of power supply regulator can have a significant impact on the complexity, cost, and performance of the power supply circuit. A power supply regulator must provide adequate power to the GTP transceiver with a minimum amount of noise while meeting the overall system thermal and efficiency requirements. There are two major types of power supply voltage regulators available for regulating the GTP analog voltage rails; linear regulators, and switching regulators. Each of these types of regulators has advantages and disadvantages. The optimal choice of regulator type depends on system requirements such as:

- Physical size
- Thermal budget
- Power efficiency
- Cost

### Linear Regulator

A linear regulator is usually the simplest means to provide voltage regulation for the GTP analog supply rails. Inherently, a linear regulator does not inject significant noise into the regulated output voltage. Not all linear regulators provide noise rejection at the output

from noise present on the voltage input. Another advantage of the linear regulator is that it usually requires a minimal number of external components to realize a circuit on the printed circuit board.

There are potentially two major disadvantages to linear regulators; minimum dropout voltage, and limited efficiency. Linear regulators require an input voltage that is higher than the output voltage. This minimum dropout voltage often is dependent on the load current. Even low dropout linear regulators require a minimum difference between the input voltage and the output voltage of the regulator. The system power supply design must consider the minimum dropout voltage requirements of the linear regulators.

The efficiency of a linear regulator is dependent on voltage difference between the input and output of the linear regulator. For instance, if the input voltage of the regulator is 2.5 VDC and the output voltage of the regulator is 1.2 V, the voltage difference is 1.3 VDC. Assuming that the current into the regulator is essentially equal to the current out of the regulator, the maximum efficiency of the regulator is 48%. This means that for every watt delivered to the load, the system must consume an additional watt for regulation. This power consumed by the regulator generates heat that must be dissipated by the system. Providing a means to dissipate the heat generated by the linear regulator can drive up the system cost. So even though from a simple component count and complexity cost the linear regulator appears to have an advantage over the switching regulator, if the overall system cost is considered including power consumption and heat dissipation, in high current applications, the linear regulator can actually be at a disadvantage.

### Switching Regulator

A switching regulator can provide a very efficient means to deliver a well regulated voltage for the GTP analog power supply. Unlike the linear regulator, the switching regulator does not depend on the voltage drop between the input voltage of the regulator and the output voltage to provide regulation. Therefore the switching regulator can supply large amounts of current to the load while maintaining high power efficiency. It is not uncommon for a switching regulator to maintain efficiencies of 95% or greater. This efficiency is not severely impacted by the voltage drop between the input of the regulator and the output. It is impacted by the load current in a much lesser degree than that of the linear regulator. Because of the efficiency of the switching regulator, the system does not need to supply as much power to the circuit and it does not need to provide a means to dissipate power consumed by the regulator.

The disadvantages to the switching regulator are complexity of the circuit and noise generated by the regulator switching function. Switching regulator circuits are usually more complex than linear regulator circuits. This shortcoming in switching regulators has recently been addressed by several switching regulator component vendors. Normally a switching power supply regulation circuit requires a switching transistor element, an inductor, and a capacitor. Depending on the required efficiency and load requirements, a switching regulator circuit may require external switching transistors and inductors. Besides the component count, these switching regulators require very careful placement and routing on the printed circuit board in order to be effective.

Switching regulators generate significant noise and therefore usually require additional filtering before the voltage is delivered to the GTP analog power supply input of the Artix-7 FPGA. As mentioned previously, the amplitude of the noise should be limited to less than 10 mV<sub>PK-PK</sub>. Therefore the power supply filter should be designed to attenuate the noise from the switching regulator so that it will meet this requirement.

## Power Supply Distribution Network

### Staged Decoupling

#### Die

There is de-coupling capacitance on the die to filter the highest frequency noise components on the power supplies. The source for this very high frequency noise will be the internal on-die circuits.

#### Printed Circuit Board

The decoupling capacitors on the printed circuit board play an important role in minimizing the effects of power supply noise on the transceivers. By providing a low impedance between the power and ground planes, the PCB decoupling capacitors provide isolation between GTP transceivers in the package.

De-coupling capacitors provide two basic functions. They help to isolate one circuit from another, and they provide isolation between the power supply source and the load circuit. By minimizing the power-to-ground impedance, noise induced on the power supply by one circuit will not induce noise on the power supply of another circuit that is sharing the same power supply. In this case, the concern would be noise coupling between GTP transceivers in the same FPGA. Also, de-coupling capacitors provide isolation between the power supply source and the load circuit.

### Power Supply Decoupling Capacitors

For the Artix-7 FPGA GTP transceiver analog power supplies, the primary purpose of decoupling capacitors is to reduce the amplitude of the noise from the power supply source and other circuits on the PCB. The suggested filtering for the power supplies, MGTAVCC and MGTAVTT, is provided in [Table 5-7](#).

**Table 5-7: Power Supply Filter Capacitor Recommendations**

Qty/Power Supply Group		Capacitance ( $\mu$ F)	Tolerance	Type
MGTAVCC	MGTAVTT			
1	1	4.7	10%	Ceramic
2	2	0.1	10%	Ceramic

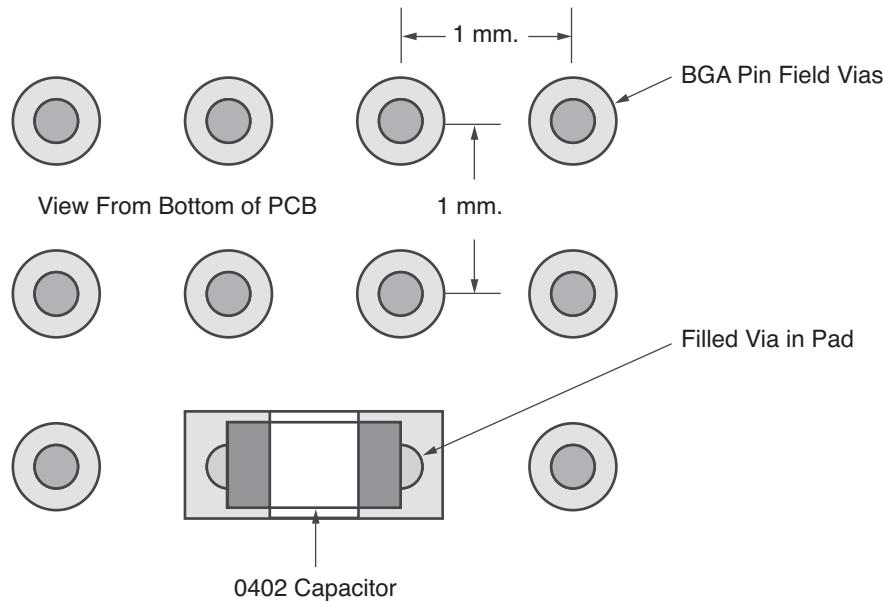
### Power Supply Decoupling Capacitor Layout and Placement

The effectiveness of the decoupling capacitors is directly dependent on their placement and routing on the printed circuit board. The inductance of the path between the capacitors and the power and ground planes on the die must be kept to a minimum. The lower the inductance in the path, the lower the voltage noise generated by transient load currents.

The larger 4.7  $\mu$ F capacitors should be placed in close proximity and outside the perimeter of the FPGA pin field.

The 0.1 $\mu$ F capacitors on the other hand must be placed as close to the GTP Quad power supply pins as possible. Placing the capacitors on the bottom of the board under the FPGA meets this requirement. A couple of options are available for placing these capacitors.

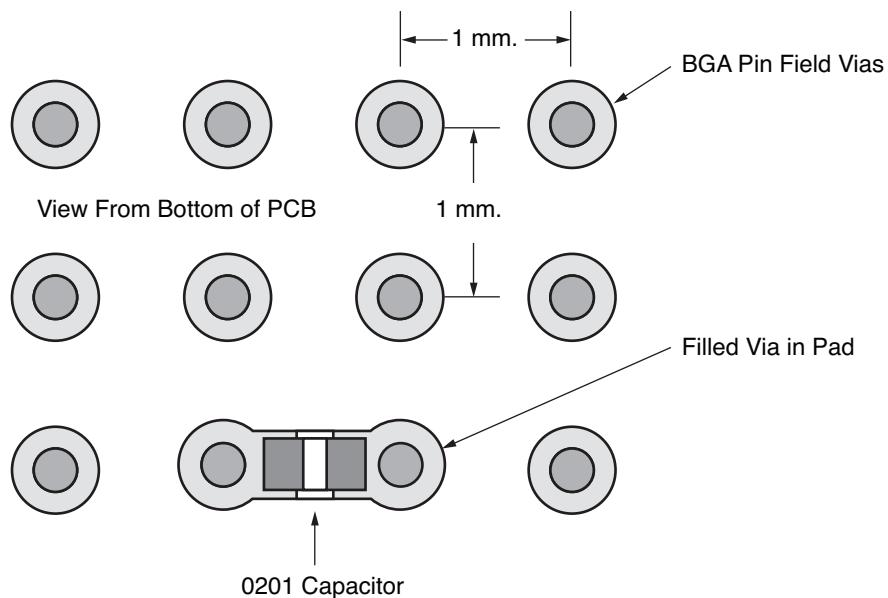
One option is to use filled vias with an 0402 size capacitor. A capacitor of this size fits between the vias. The via hole needs to be filled to keep the solder from wicking into the via. See [Figure 5-10](#) for an example of placement and routing of the 0.1  $\mu$ F capacitors.



UG482\_c5\_10\_072412

Figure 5-10: Placement of 0.1  $\mu\text{F}$  0402 Capacitor Using Filled Via in Pad Under FPGA

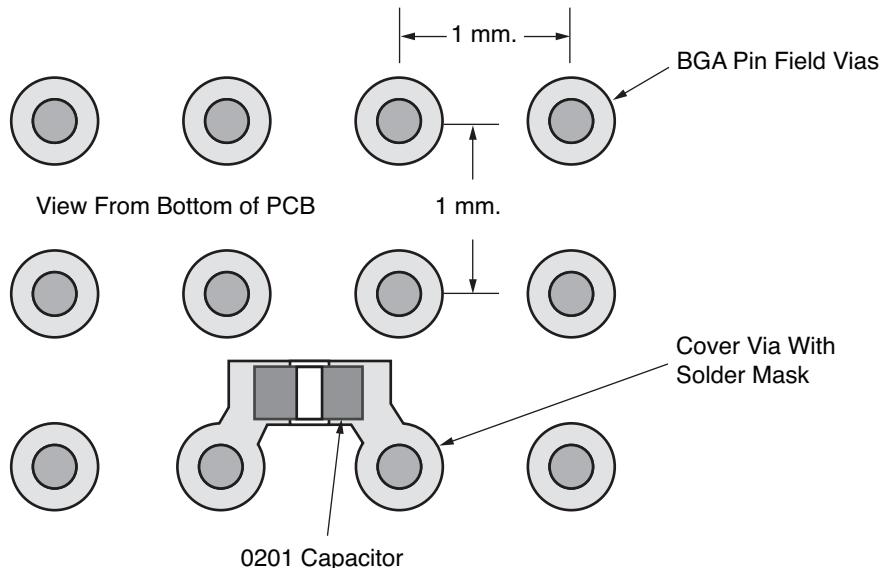
Another option is to use an 0201 capacitor and mounting it between the vias. An example of this is shown in Figure 5-11.



UG482\_c5\_11\_072412

Figure 5-11: Placement of 0.1  $\mu\text{F}$  0201 Capacitor Using Filled Via in Pad Under FPGA

Another choice is to mount an 0201 capacitor next to the BGA via pads. Doing this eliminates the need to use filled via in pad. An example of this placement is shown in Figure 5-12.



UG482\_c5\_12\_072412

**Figure 5-12: Placement of 0.1  $\mu$ F 0201 Capacitor on Bottom of PCB Under FPGA**

The Artix-7 FPGA and Zynq-7000 AP SoC packages with GTP transceivers have analog power supply pins and adjacent ground pins. Table 5-8 through Table 5-13 show suggested power and ground pin pairs for mounting the 0.1  $\mu$ F capacitors. Following the guidance in these tables and using the previously discussed layout and placement guidance provides effective power supply decoupling by maintaining a minimum amount of inductance in the path between the on-die circuitry and the capacitor on the printed circuit board.

**Table 5-8: CLG485 Package – 0.1  $\mu$ F Capacitor Placement**

Capacitor	Package Pins			Value
	MGTAVCC	MGTAVTT	GND	
Cap1		Y7	W7	0.1 $\mu$ F
Cap2		AA4	AB4	
Cap3	U8		V8	
Cap4	W5		Y5	

**Table 5-9: CPG236 Package – 0.1  $\mu$ F Capacitor Placement**

Capacitor	Package Pins			Value
	MGTAVCC	MGTAVTT	GND	
Cap1		G7	G8	0.1 $\mu$ F
Cap2		B1	A1	

Table 5-9: CPG236 Package – 0.1 µF Capacitor Placement (Cont'd)

Capacitor	Package Pins			Value
	MGTAVCC	MGTAVTT	GND	
Cap3	H9		H8	0.1 µF
Cap4	C1		C2	

Table 5-10: CSG325 Package – 0.1 µF Capacitor Placement

Capacitor	Package Pins			Value
	MGTAVCC	MGTAVTT	GND	
Cap1		F3	F4	0.1 µF
Cap2		A2	A1	
Cap3	B4		B3	
Cap4	F5		F6	

Table 5-11: FGG484/FBG484/SBG484 Package – 0.1 µF Capacitor Placement

Capacitor	Package Pins			Value
	MGTAVCC	MGTAVTT	GND	
Cap1	D6	-	C6	0.1 µF
Cap2	D10	-	C10	
Cap3	-	C4	C3	
Cap4	-	B11	B12	

Table 5-12: FGG676/FBG676 Package – 0.1 µF Capacitor Placement

Capacitor	Power Supply Group	Package Pins			Value
		MGTAVCC	MGTAVTT	GND	
Cap1	G10	AC9	-	AD9	0.1 µF
Cap2		AA12	-	AB12	
Cap5		-	AD7	AC7	
Cap6		-	AD15	AC15	
Cap3	G11	D9	-	E9	
Cap4		D13	-	C13	
Cap7		-	C7	D7	
Cap8		-	C15	D15	

Table 5-13: FFG1156 Package – 0.1  $\mu$ F Capacitor Placement

Capacitor	Power Supply Group	Package Pins			Value
		MGTAVCC	MGTAVTT	GND	
Cap1	G10	AJ14	-	AK14	0.1 $\mu$ F
Cap2		AJ20	-	AK20	
Cap5		-	AN14	AP14	
Cap6		-	AN22	AP22	
Cap3	G11	F14	-	E14	
Cap4		F20	-	E20	
Cap7		-	B14	A14	
Cap8		-	B22	A22	

## Crosstalk

A major contributor to degradation in the performance of an MGT is crosstalk. The mechanisms for crosstalk are aggressor signals coupling into signal traces and/or coupling into the MGT power supplies. The latter is the most common and often the most damaging. Noise coupled into the power supply can corrupt the entire transceiver circuit rather than just a single lane as in the case of coupling into signal traces. Also, the effect of noise coupled into the power supply is that the symptoms are often more difficult to interpret because the noise is convolved with the normal signals in the transceiver. The result is degradation in the performance of the transceiver that reveals itself as noise in the transmitter output and reduced jitter tolerance in the receiver.

To avoid performance degradation from crosstalk:

- Monitor the exposure of power planes to other circuits on the board, such as data lines for memory interfaces and processor busses.
- Provide adequate filtering to the MGT power supplies near the point of the load. The amount of filtering should be determined by the magnitude and frequency of the signal from potential noise sources. Noise on the MGT power supplies should be kept below 10 mV<sub>PK-PK</sub> from 10 kHz to 80 MHz.
- Be aware of the return current paths of signal traces in the vicinity of the MGT power distribution network. Besides broadband and edge coupling of traces on the same or adjacent layers, coupling from aggressor traces can occur if the aggressor signal is propagating from one layer to another each layer having a different reference plane. As the signal propagates through the portion of the via that does not have a return current path, it will generate return currents in the next lowest impedance structure on the board. That victim could be a signal or power via for the MGTs.

## SelectIO Usage Guidelines

Because a GTP transceiver's performance can degrade in an environment flooded with SelectIOTM activity, it is important to have guidelines for SelectIO usage that minimize the impact on GTP transceiver performance.

The following guidelines must be followed when routing GTP Transceiver data signals on the PCB:

- Eliminate routing of GTP Transceiver signals and SelectIO signals on adjacent layers. Be aware of the potential of broadside coupling if these signals are routed on adjacent layers.
- Maintain isolation of the return current paths for both the SelectIO signals and the GTP Transceiver signals including both traces and vias.
- The power islands for the GTP Transceivers are also a potential source for SelectIO induced noise. SelectIO signals should not be routed over the GTP power islands.

### Specific SelectIO Guidelines Pertaining to the FGG676 Package

- To minimize the impact to GTP performance from SelectIO in adjacent banks, SelectIO banks 16 and 35 should be avoided for line rates equal to or above 6 Gb/s.
- If SelectIO banks 16 and 35 must be used, then it is recommended to reduce the number of SelectIOs used in these banks and the following selectIOs are prohibited:
  - Bank 16: F17, F18, F20, G15, H14, H15, A17, A18, A19, B17, B19, C17, D16, D18, E16, E18, F15, F19
  - Bank 35: K8, J8, J6, J5, J4, H9, H8, H7, H6, H4, G9, G8, G7, G6, F8, F7, E6, D6

## PCB Design Checklist

[Table 5-14](#) is a checklist of items that can be used to design and review any 7-Series GTP transceiver PCB schematic and layout.

*Table 5-14: GTP PCB Design Checklist*

Pins	Recommendations
MGTREFCLK0P MGTREFCLK0N MGTREFCLK1P MGTREFCLK1N	<ul style="list-style-type: none"> <li>• Use AC coupling capacitors for connection to oscillator.</li> <li>• For AC coupling capacitors, refer to <a href="#">Reference Clock, page 230</a>. The recommended value for LVDS is 100 nF.</li> <li>• Reference clock traces should be provided enough clearance to eliminate crosstalk from adjacent signals.</li> <li>• Reference clock oscillator output must comply with the min/max input amplitude requirements for these input pins. See <a href="#">DS181, Artix-7 FPGAs Data Sheet: DC and Switching Characteristics</a>.</li> <li>• If reference clock input is not used leave the associated pin pair unconnected.</li> </ul>
MGTRXP0/MGTRXN0 MGTRXP1/MGTRXN1 MGTRXP2/MGTRXN2 MGTRXP3/MGTRXN3	<ul style="list-style-type: none"> <li>• Use AC coupling capacitors for connection to transmitter. The recommended value for AC coupling capacitors is 100 nF.</li> <li>• Receiver data traces should be provided enough clearance to eliminate crosstalk from adjacent signals.</li> <li>• If a receiver is not used connect the associated pin pair to ground.</li> <li>• See <a href="#">RX Analog Front End, page 128</a>.</li> </ul>

Table 5-14: GTP PCB Design Checklist (Cont'd)

Pins	Recommendations
MGTTXP0/MGTTXN0 MGTTXP1/MGTTXN1 MGTTXP2/MGTTXN2 MGTTXP3/MGTTXN3	<ul style="list-style-type: none"> <li>Transmitter should be AC coupled to the receiver. The recommended value for the AC coupling capacitors is 100 nF.</li> <li>Transmitter data traces should be provided enough clearance to eliminate crosstalk from adjacent signals.</li> <li>If a transmitter is not used leave the associated pin pair unconnected.</li> </ul>
MGTRREF	<ul style="list-style-type: none"> <li>Connect to a <math>100\Omega</math> resistor that is also connected to MGTAVTT.</li> <li>See <a href="#">Termination Resistor Calibration Circuit, page 225</a>.</li> </ul>
MGTAVCC_G[N]	<ul style="list-style-type: none"> <li>The nominal voltage is 1.0 VDC.</li> <li>See <a href="#">DS181, Artix-7 FPGAs Data Sheet: DC and Switching Characteristics</a> for power supply voltage tolerances.</li> <li>The power supply regulator for this voltage should not be shared with non-transceiver loads.</li> <li>Many packages have multiple groups of power supply connections in the package for MGTAVCC. Refer to <a href="#">Table 5-2</a> to identify in which power supply group a specific GTP Quad is located. Information on pin locations for each package can be found in <a href="#">UG475, 7 Series FPGAs Packaging and Pinout Specifications</a>.</li> <li>The following sets of ceramic filter capacitors for each power supply group are recommended: <ul style="list-style-type: none"> <li>1 of <math>4.7\ \mu\text{F}</math> 10%</li> <li>2 of <math>0.1\ \mu\text{F}</math> 10%</li> </ul> </li> <li>For optimal performance, power supply noise must be less than <math>10\ \text{mV}_{\text{PK-PK}}</math>.</li> <li>If all of the Quads in a power supply group are not used, the associated power pins can be tied to ground.</li> <li>For power consumption refer to the XPower Estimator (XPE) for 7 series devices at <a href="http://www.xilinx.com/power">www.xilinx.com/power</a>.</li> </ul>
MGTAVTT_G[N]	<ul style="list-style-type: none"> <li>The nominal voltage is 1.2 VDC.</li> <li>See <a href="#">DS181, Artix-7 FPGAs Data Sheet: DC and Switching Characteristics</a> for power supply voltage tolerances.</li> <li>The power supply regulator for this voltage should not be shared with non-MGT loads.</li> <li>Many packages have multiple groups of power supply connections in the package for MGTAVTT. Refer to <a href="#">Table 5-2</a> to identify in which power supply group a specific GTP Quad is located. Information on pin locations for each package can be found in <a href="#">UG475, 7 Series FPGAs Packaging and Pinout Specifications</a>.</li> <li>The following set of ceramic filter capacitors for each power supply group are recommended: <ul style="list-style-type: none"> <li>1 of <math>4.7\ \mu\text{F}</math> 10%</li> <li>2 of <math>0.1\ \mu\text{F}</math> 10%</li> </ul> </li> <li>For optimal performance power supply noise must be less than <math>10\ \text{mV}_{\text{PK-PK}}</math>.</li> <li>If all of the Quads in a power supply group are not used the associated power pins can be tied to ground.</li> <li>For power consumption refer to XPower Estimator (XPE) for 7 series devices at <a href="http://www.xilinx.com/power">www.xilinx.com/power</a>.</li> </ul>

## *Placement Information by Package*

---

This appendix provides the Quad position information for available device and package combinations along with the pad numbers for the external signals associated with each serial transceiver channel and the associated primitive. References to the XC part numbers also apply to the XA and XQ part numbers, where available.

- [CPG236 Package Placement Diagram, page 244](#)
- [CSG325 Package Placement Diagram, page 245](#)
- [CLG485 Package Placement Diagram, page 246](#)
- [FGG484 Package Placement Diagram, page 247](#)
- [FGG676 Package Placement Diagram, page 248](#)
- [FBG484 Package Placement Diagram, page 250](#)
- [SBG484 Package Placement Diagram, page 251](#)
- [FBG676 Package Placement Diagram, page 252](#)
- [FFG1156 Package Placement Diagram, page 254](#)

## CPG236 Package Placement Diagram

Figure A-1 show the placement diagram for the CPG236 package.

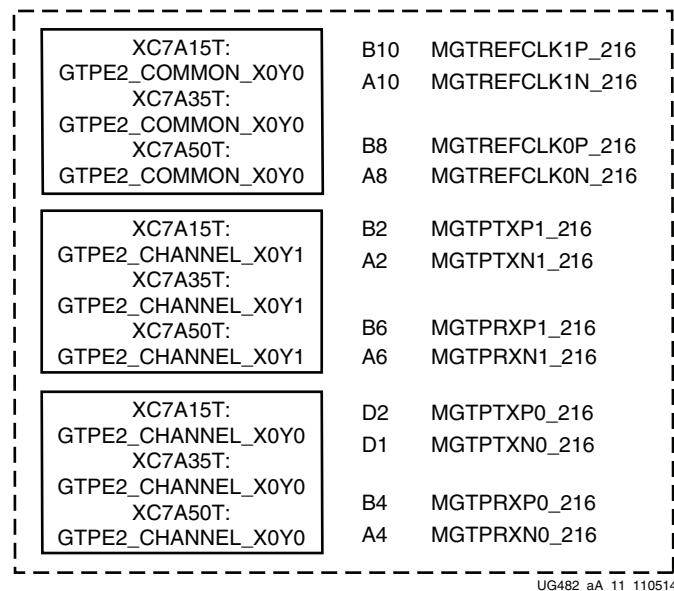


Figure A-1: Placement Diagram for the CPG236 Package

## CSG325 Package Placement Diagram

Figure A-2 and shows the placement diagram for the CSG325 package.

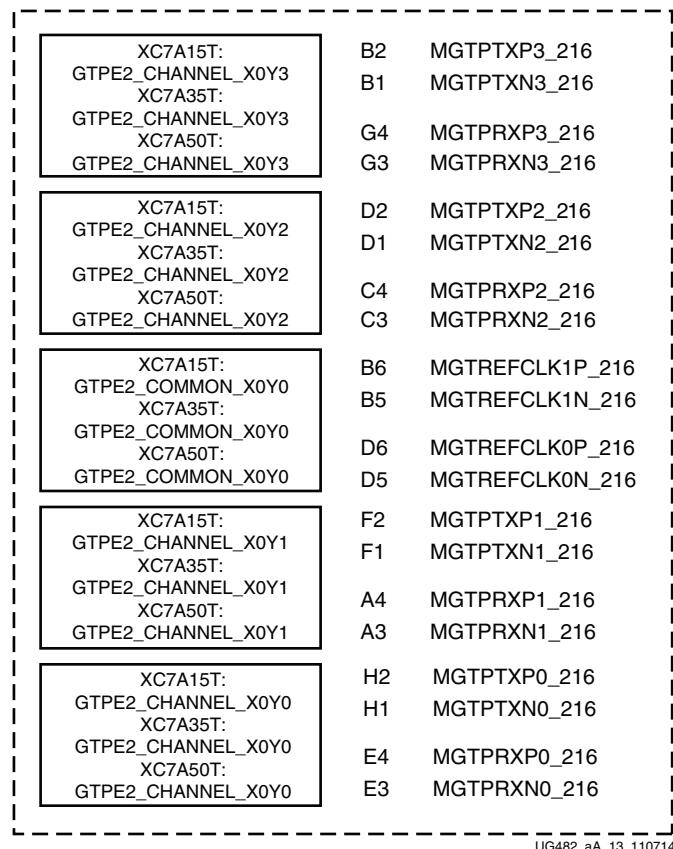


Figure A-2: Placement Diagram for the CSG325 Package

## CLG485 Package Placement Diagram

Figure A-3 and shows the placement diagram for the CLG485 package.

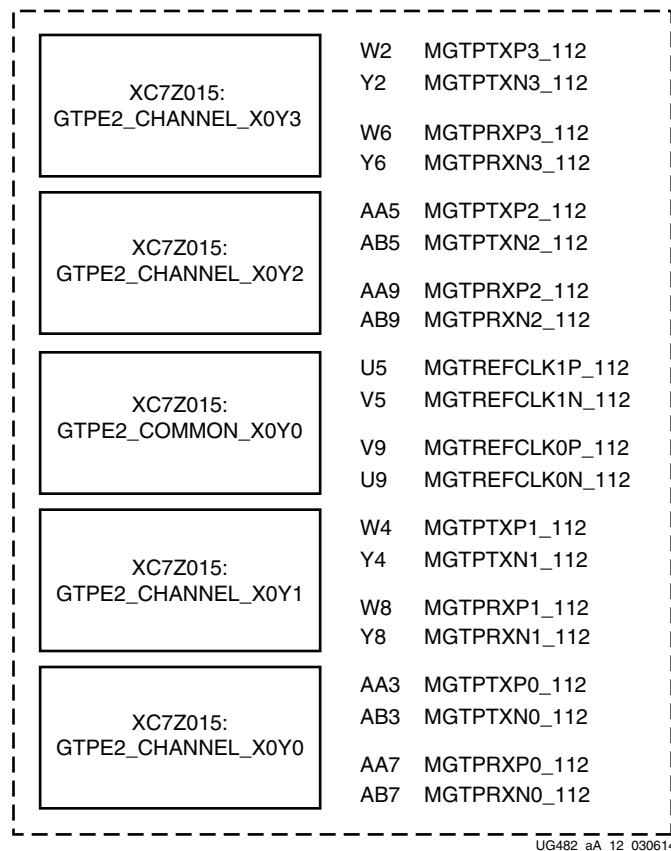


Figure A-3: Placement Diagram for the CLG485 Package

## FGG484 Package Placement Diagram

Figure A-4 and shows the placement diagram for the FGG484 package.

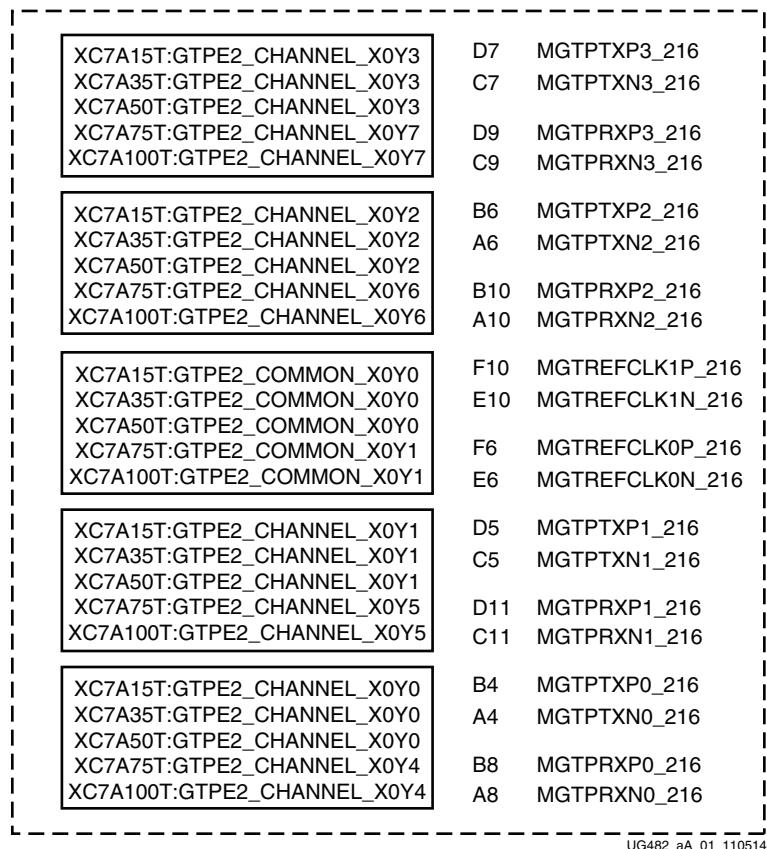


Figure A-4: Placement Diagram for the FGG484 Package

## FGG676 Package Placement Diagram

Figure A-5 and Figure A-6 show the placement diagram for the FGG676 package.

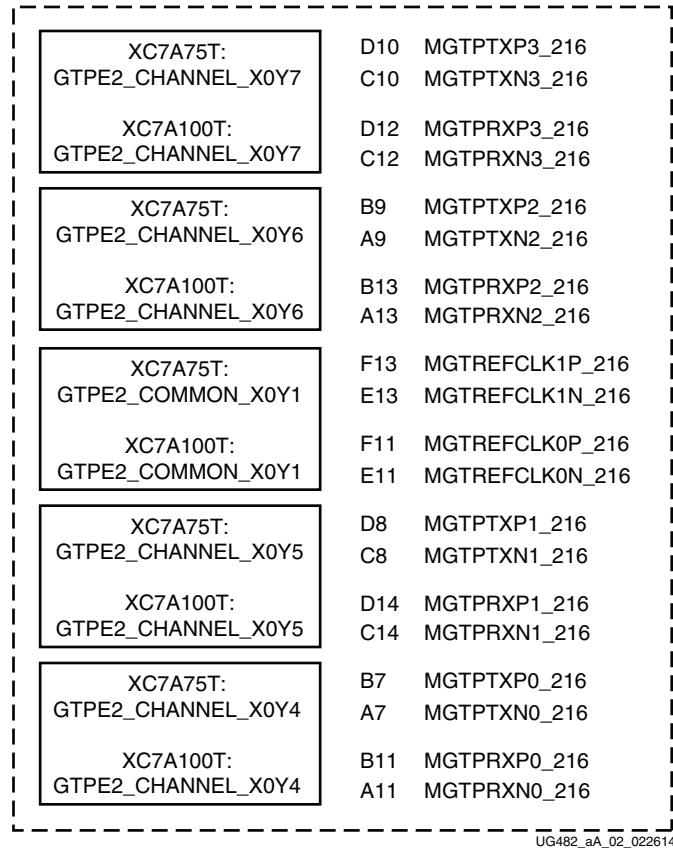


Figure A-5: Placement Diagram for the FGG676 Package (1 of 2)

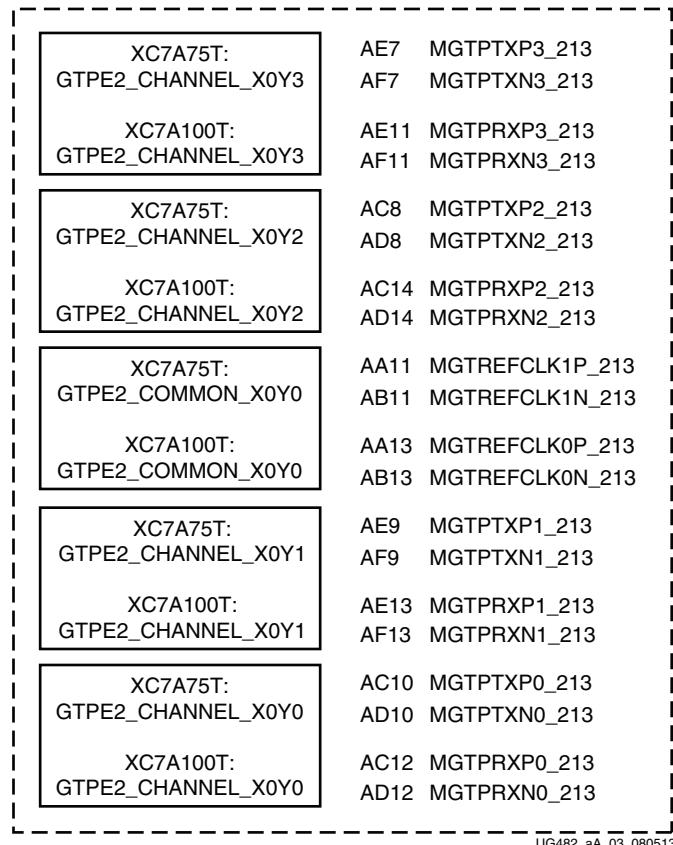


Figure A-6: Placement Diagram for the FGG676 Package (2 of 2)

## FBG484 Package Placement Diagram

Figure A-7 show the placement diagram for the FBG484 package.

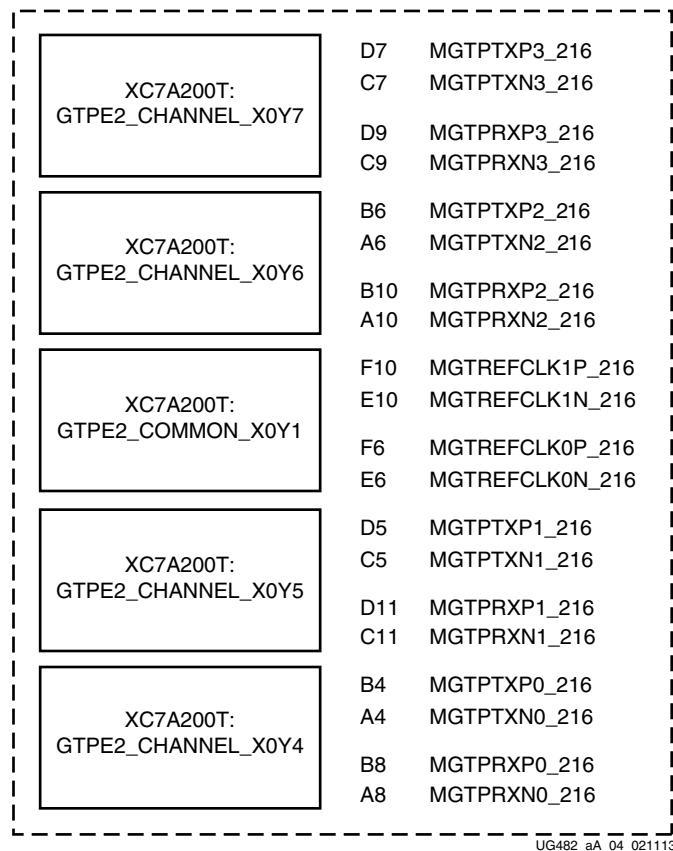


Figure A-7: Placement Diagram for the FBG484 Package

## SBG484 Package Placement Diagram

Figure A-8 show the placement diagram for the SBG484 package.

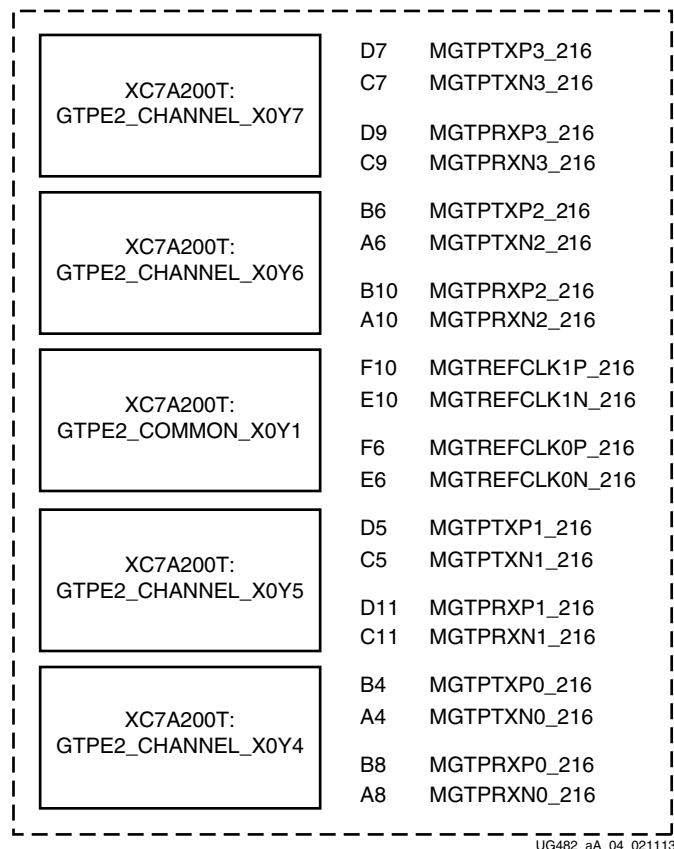


Figure A-8: Placement Diagram for the SBG484 Package

## FBG676 Package Placement Diagram

Figure A-9 and Figure A-10 show the placement diagram for the FBG676 package.

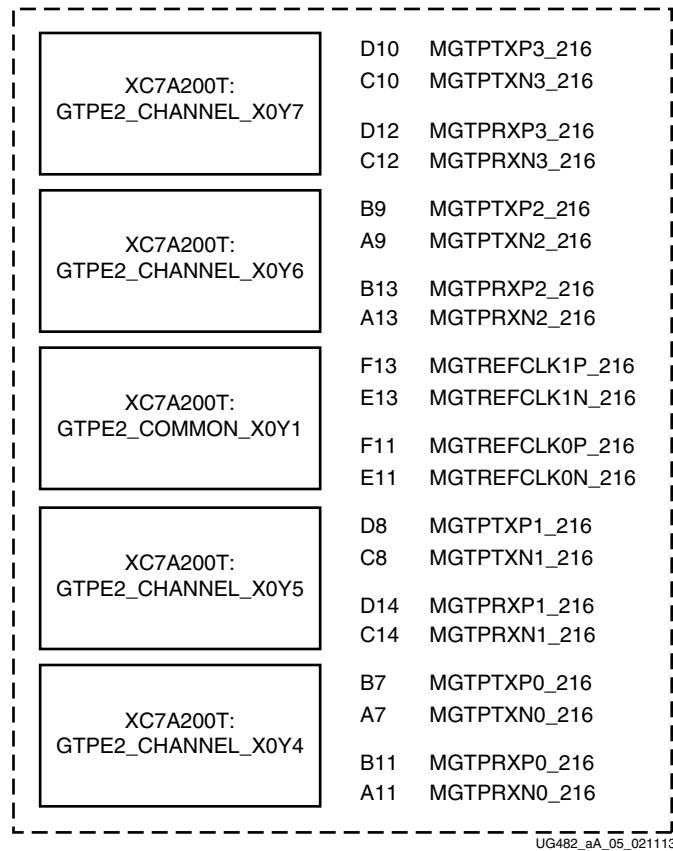


Figure A-9: Placement Diagram for the FBG676 Package (1 of 2)

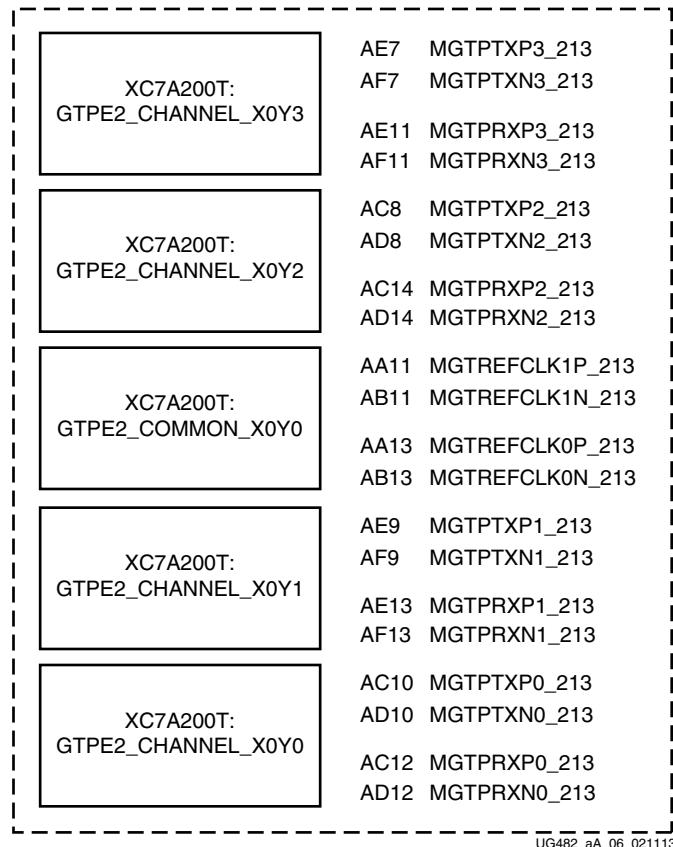


Figure A-10: Placement Diagram for the FBG676 Package (2 of 2)

## FFG1156 Package Placement Diagram

Figure A-11 through Figure A-14 show the placement diagram for the FFG1156 package.

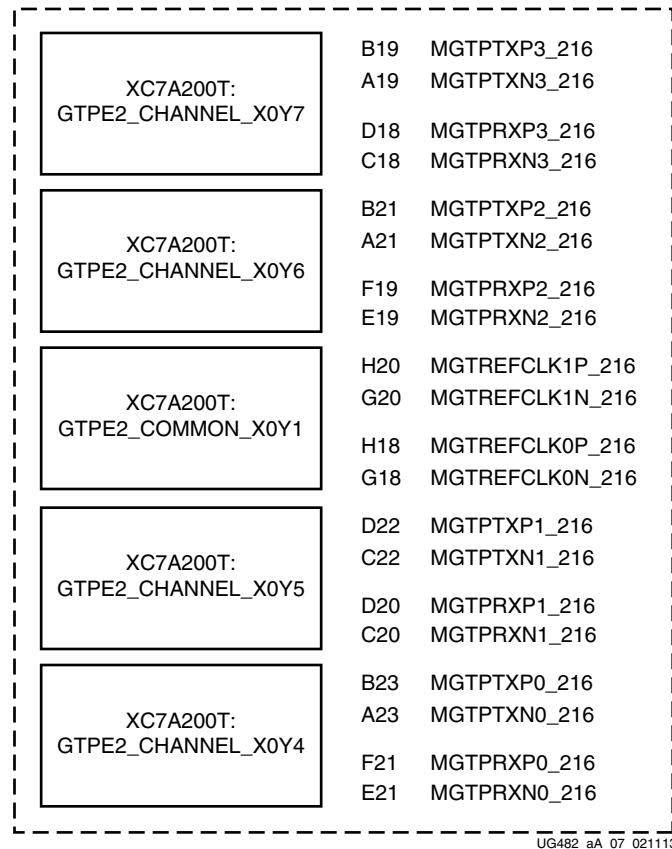


Figure A-11: Placement Diagram for the FFG1156 Package (1 of 4)

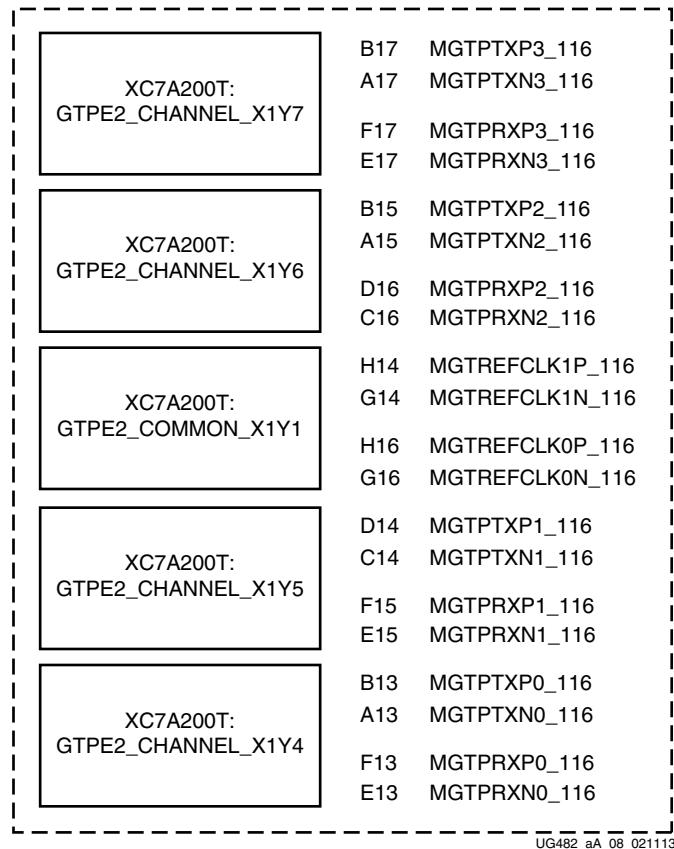


Figure A-12: Placement Diagram for the FFG1156 Package (2 of 4)

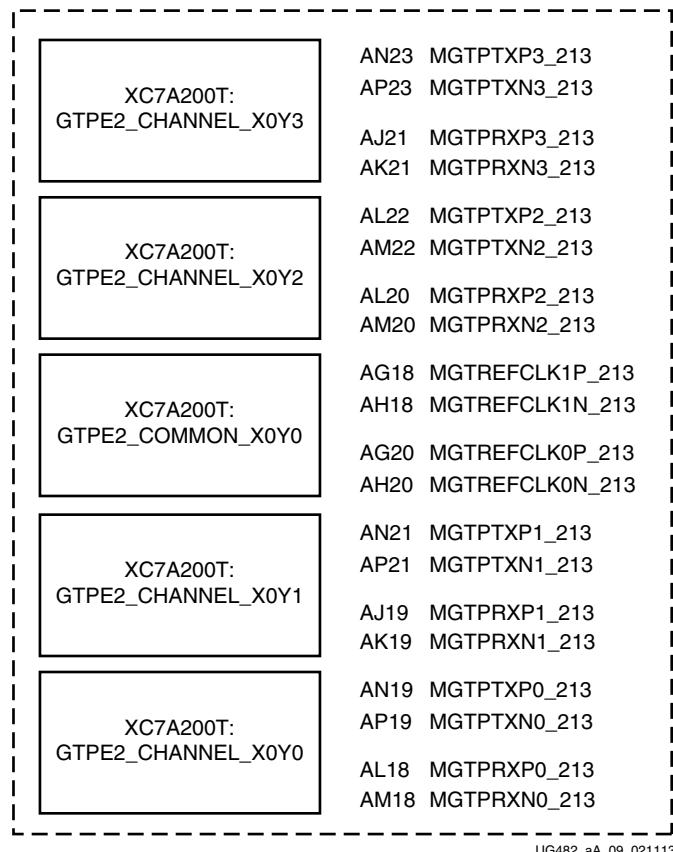


Figure A-13: Placement Diagram for the FFG1156 Package (3 of 4)

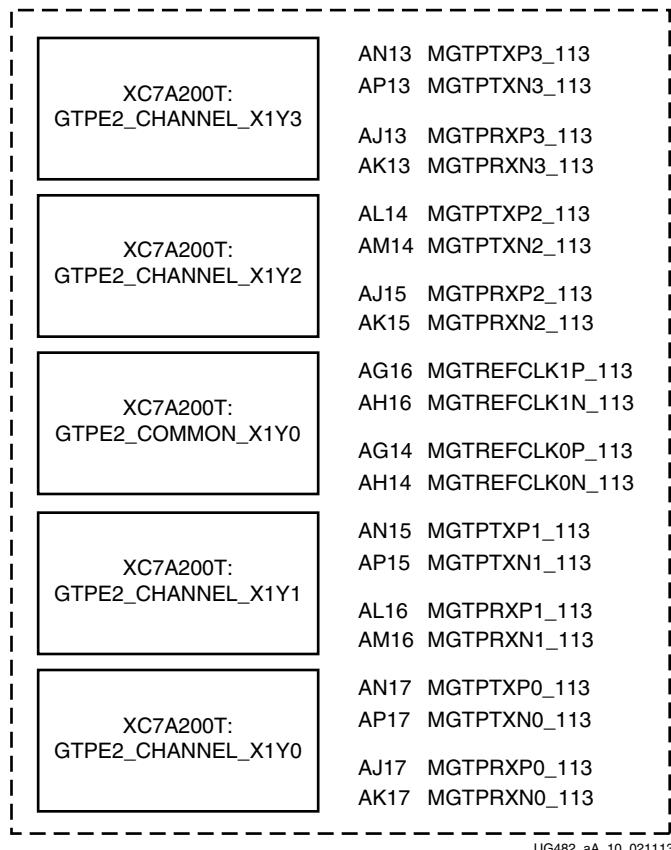


Figure A-14: Placement Diagram for the FFG1156 Package (4 of 4)



# *Placement Information by Device*

---

[Table B-1](#) defines the Artix™-7 FPGA device-package combinations and the available GTP transceiver banks. [Table B-2](#) defines the Zynq-7000 device-package combinations and the available GTP transceiver banks. For transceiver location, refer to [Appendix A, Placement Information by Package](#).

**Table B-1: Artix-7 FPGA Device-Package Combinations and GTP Transceiver Banks**

Package	CPG236	CSG325	FGG484	FGG676	SBG484	FBG484	FBG676	FFG1156
XC7A15T	MGT_BANK_216	MGT_BANK_216	MGT_BANK_216					
XC7A35T	MGT_BANK_216	MGT_BANK_216	MGT_BANK_216					
XC7A50T	MGT_BANK_216	MGT_BANK_216	MGT_BANK_216					
XC7A75T			MGT_BANK_216	MGT_BANK_213, MGT_BANK_216				
XC7A100T			MGT_BANK_216	MGT_BANK_213, MGT_BANK_216				
XC7A200T					MGT_BANK_216	MGT_BANK_213, MGT_BANK_216	MGT_BANK_113, MGT_BANK_116, MGT_BANK_213, MGT_BANK_216	

**Table B-2: Zynq-7000 Device/Package Combinations and GTP Transceiver Banks**

Package	CLG485
XC7Z015	MGT_BANK_112



## 8B/10B Valid Characters

---

8B/10B encoding includes a set of Data characters and K characters. Eight-bit values are coded into 10-bit values, keeping the serial line DC balanced. K characters are special Data characters designated with a CHARISK. K characters are used for specific informative designations. [Table C-1](#) shows the valid Data characters. [Table C-2, page 269](#) shows the valid K characters.

**Table C-1: Valid Data Characters**

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D0.0	000 00000	100111 0100	011000 1011
D1.0	000 00001	011101 0100	100010 1011
D2.0	000 00010	101101 0100	010010 1011
D3.0	000 00011	110001 1011	110001 0100
D4.0	000 00100	110101 0100	001010 1011
D5.0	000 00101	101001 1011	101001 0100
D6.0	000 00110	011001 1011	011001 0100
D7.0	000 00111	111000 1011	000111 0100
D8.0	000 01000	111001 0100	000110 1011
D9.0	000 01001	100101 1011	100101 0100
D10.0	000 01010	010101 1011	010101 0100
D11.0	000 01011	110100 1011	110100 0100
D12.0	000 01100	001101 1011	001101 0100
D13.0	000 01101	101100 1011	101100 0100
D14.0	000 01110	011100 1011	011100 0100
D15.0	000 01111	010111 0100	101000 1011
D16.0	000 10000	011011 0100	100100 1011
D17.0	000 10001	100011 1011	100011 0100
D18.0	000 10010	010011 1011	010011 0100
D19.0	000 10011	110010 1011	110010 0100
D20.0	000 10100	001011 1011	001011 0100

Table C-1: Valid Data Characters (Cont'd)

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D21.0	000 10101	101010 1011	101010 0100
D22.0	000 10110	011010 1011	011010 0100
D23.0	000 10111	111010 0100	000101 1011
D24.0	000 11000	110011 0100	001100 1011
D25.0	000 11001	100110 1011	100110 0100
D26.0	000 11010	010110 1011	010110 0100
D27.0	000 11011	110110 0100	001001 1011
D28.0	000 11100	001110 1011	001110 0100
D29.0	000 11101	101110 0100	010001 1011
D30.0	000 11110	011110 0100	100001 1011
D31.0	000 11111	101011 0100	010100 1011
D0.1	001 00000	100111 1001	011000 1001
D1.1	001 00001	011101 1001	100010 1001
D2.1	001 00010	101101 1001	010010 1001
D3.1	001 00011	110001 1001	110001 1001
D4.1	001 00100	110101 1001	001010 1001
D5.1	001 00101	101001 1001	101001 1001
D6.1	001 00110	011001 1001	011001 1001
D7.1	001 00111	111000 1001	000111 1001
D8.1	001 01000	111001 1001	000110 1001
D9.1	001 01001	100101 1001	100101 1001
D10.1	001 01010	010101 1001	010101 1001
D11.1	001 01011	110100 1001	110100 1001
D12.1	001 01100	001101 1001	001101 1001
D13.1	001 01101	101100 1001	101100 1001
D14.1	001 01110	011100 1001	011100 1001
D15.1	001 01111	010111 1001	101000 1001
D16.1	001 10000	011011 1001	100100 1001
D17.1	001 10001	100011 1001	100011 1001
D18.1	001 10010	010011 1001	010011 1001
D19.1	001 10011	110010 1001	110010 1001
D20.1	001 10100	001011 1001	001011 1001

Table C-1: Valid Data Characters (Cont'd)

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D21.1	001 10101	101010 1001	101010 1001
D22.1	001 10110	011010 1001	011010 1001
D23.1	001 10111	111010 1001	000101 1001
D24.1	001 11000	110011 1001	001100 1001
D25.1	001 11001	100110 1001	100110 1001
D26.1	001 11010	010110 1001	010110 1001
D27.1	001 11011	110110 1001	001001 1001
D28.1	001 11100	001110 1001	001110 1001
D29.1	001 11101	101110 1001	010001 1001
D30.1	001 11110	011110 1001	100001 1001
D31.1	001 11111	101011 1001	010100 1001
D0.2	010 00000	100111 0101	011000 0101
D1.2	010 00001	011101 0101	100010 0101
D2.2	010 00010	101101 0101	010010 0101
D3.2	010 00011	110001 0101	110001 0101
D4.2	010 00100	110101 0101	001010 0101
D5.2	010 00101	101001 0101	101001 0101
D6.2	010 00110	011001 0101	011001 0101
D7.2	010 00111	111000 0101	000111 0101
D8.2	010 01000	111001 0101	000110 0101
D9.2	010 01001	100101 0101	100101 0101
D10.2	010 01010	010101 0101	010101 0101
D11.2	010 01011	110100 0101	110100 0101
D12.2	010 01100	001101 0101	001101 0101
D13.2	010 01101	101100 0101	101100 0101
D14.2	010 01110	011100 0101	011100 0101
D15.2	010 01111	010111 0101	101000 0101
D16.2	010 10000	011011 0101	100100 0101
D17.2	010 10001	100011 0101	100011 0101
D18.2	010 10010	010011 0101	010011 0101
D19.2	010 10011	110010 0101	110010 0101
D20.2	010 10100	001011 0101	001011 0101

Table C-1: Valid Data Characters (Cont'd)

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D21.2	010 10101	101010 0101	101010 0101
D22.2	010 10110	011010 0101	011010 0101
D23.2	010 10111	111010 0101	000101 0101
D24.2	010 11000	110011 0101	001100 0101
D25.2	010 11001	100110 0101	100110 0101
D26.2	010 11010	010110 0101	010110 0101
D27.2	010 11011	110110 0101	001001 0101
D28.2	010 11100	001110 0101	001110 0101
D29.2	010 11101	101110 0101	010001 0101
D30.2	010 11110	011110 0101	100001 0101
D31.2	010 11111	101011 0101	010100 0101
D0.3	011 00000	100111 0011	011000 1100
D1.3	011 00001	011101 0011	100010 1100
D2.3	011 00010	101101 0011	010010 1100
D3.3	011 00011	110001 1100	110001 0011
D4.3	011 00100	110101 0011	001010 1100
D5.3	011 00101	101001 1100	101001 0011
D6.3	011 00110	011001 1100	011001 0011
D7.3	011 00111	111000 1100	000111 0011
D8.3	011 01000	111001 0011	000110 1100
D9.3	011 01001	100101 1100	100101 0011
D10.3	011 01010	010101 1100	010101 0011
D11.3	011 01011	110100 1100	110100 0011
D12.3	011 01100	001101 1100	001101 0011
D13.3	011 01101	101100 1100	101100 0011
D14.3	011 01110	011100 1100	011100 0011
D15.3	011 01111	010111 0011	101000 1100
D16.3	011 10000	011011 0011	100100 1100
D17.3	011 10001	100011 1100	100011 0011
D18.3	011 10010	010011 1100	010011 0011
D19.3	011 10011	110010 1100	110010 0011
D20.3	011 10100	001011 1100	001011 0011

Table C-1: Valid Data Characters (Cont'd)

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D21.3	011 10101	101010 1100	101010 0011
D22.3	011 10110	011010 1100	011010 0011
D23.3	011 10111	111010 0011	000101 1100
D24.3	011 11000	110011 0011	001100 1100
D25.3	011 11001	100110 1100	100110 0011
D26.3	011 11010	010110 1100	010110 0011
D27.3	011 11011	110110 0011	001001 1100
D28.3	011 11100	001110 1100	001110 0011
D29.3	011 11101	101110 0011	010001 1100
D30.3	011 11110	011110 0011	100001 1100
D31.3	011 11111	101011 0011	010100 1100
D0.4	100 00000	100111 0010	011000 1101
D1.4	100 00001	011101 0010	100010 1101
D2.4	100 00010	101101 0010	010010 1101
D3.4	100 00011	110001 1101	110001 0010
D4.4	100 00100	110101 0010	001010 1101
D5.4	100 00101	101001 1101	101001 0010
D6.4	100 00110	011001 1101	011001 0010
D7.4	100 00111	111000 1101	000111 0010
D8.4	100 01000	111001 0010	000110 1101
D9.4	100 01001	100101 1101	100101 0010
D10.4	100 01010	010101 1101	010101 0010
D11.4	100 01011	110100 1101	110100 0010
D12.4	100 01100	001101 1101	001101 0010
D13.4	100 01101	101100 1101	101100 0010
D14.4	100 01110	011100 1101	011100 0010
D15.4	100 01111	010111 0010	101000 1101
D16.4	100 10000	011011 0010	100100 1101
D17.4	100 10001	100011 1101	100011 0010
D18.4	100 10010	010011 1101	010011 0010
D19.4	100 10011	110010 1101	110010 0010
D20.4	100 10100	001011 1101	001011 0010

Table C-1: Valid Data Characters (Cont'd)

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D21.4	100 10101	101010 1101	101010 0010
D22.4	100 10110	011010 1101	011010 0010
D23.4	100 10111	111010 0010	000101 1101
D24.4	100 11000	110011 0010	001100 1101
D25.4	100 11001	100110 1101	100110 0010
D26.4	100 11010	010110 1101	010110 0010
D27.4	100 11011	110110 0010	001001 1101
D28.4	100 11100	001110 1101	001110 0010
D29.4	100 11101	101110 0010	010001 1101
D30.4	100 11110	011110 0010	100001 1101
D31.4	100 11111	101011 0010	010100 1101
D0.5	101 00000	100111 1010	011000 1010
D1.5	101 00001	011101 1010	100010 1010
D2.5	101 00010	101101 1010	010010 1010
D3.5	101 00011	110001 1010	110001 1010
D4.5	101 00100	110101 1010	001010 1010
D5.5	101 00101	101001 1010	101001 1010
D6.5	101 00110	011001 1010	011001 1010
D7.5	101 00111	111000 1010	000111 1010
D8.5	101 01000	111001 1010	000110 1010
D9.5	101 01001	100101 1010	100101 1010
D10.5	101 01010	010101 1010	010101 1010
D11.5	101 01011	110100 1010	110100 1010
D12.5	101 01100	001101 1010	001101 1010
D13.5	101 01101	101100 1010	101100 1010
D14.5	101 01110	011100 1010	011100 1010
D15.5	101 01111	010111 1010	101000 1010
D16.5	101 10000	011011 1010	100100 1010
D17.5	101 10001	100011 1010	100011 1010
D18.5	101 10010	010011 1010	010011 1010
D19.5	101 10011	110010 1010	110010 1010
D20.5	101 10100	001011 1010	001011 1010

Table C-1: Valid Data Characters (Cont'd)

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D21.5	101 10101	101010 1010	101010 1010
D22.5	101 10110	011010 1010	011010 1010
D23.5	101 10111	111010 1010	000101 1010
D24.5	101 11000	110011 1010	001100 1010
D25.5	101 11001	100110 1010	100110 1010
D26.5	101 11010	010110 1010	010110 1010
D27.5	101 11011	110110 1010	001001 1010
D28.5	101 11100	001110 1010	001110 1010
D29.5	101 11101	101110 1010	010001 1010
D30.5	101 11110	011110 1010	100001 1010
D31.5	101 11111	101011 1010	010100 1010
D0.6	110 00000	100111 0110	011000 0110
D1.6	110 00001	011101 0110	100010 0110
D2.6	110 00010	101101 0110	010010 0110
D3.6	110 00011	110001 0110	110001 0110
D4.6	110 00100	110101 0110	001010 0110
D5.6	110 00101	101001 0110	101001 0110
D6.6	110 00110	011001 0110	011001 0110
D7.6	110 00111	111000 0110	000111 0110
D8.6	110 01000	111001 0110	000110 0110
D9.6	110 01001	100101 0110	100101 0110
D10.6	110 01010	010101 0110	010101 0110
D11.6	110 01011	110100 0110	110100 0110
D12.6	110 01100	001101 0110	001101 0110
D13.6	110 01101	101100 0110	101100 0110
D14.6	110 01110	011100 0110	011100 0110
D15.6	110 01111	010111 0110	101000 0110
D16.6	110 10000	011011 0110	100100 0110
D17.6	110 10001	100011 0110	100011 0110
D18.6	110 10010	010011 0110	010011 0110
D19.6	110 10011	110010 0110	110010 0110
D20.6	110 10100	001011 0110	001011 0110

Table C-1: Valid Data Characters (Cont'd)

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D21.6	110 10101	101010 0110	101010 0110
D22.6	110 10110	011010 0110	011010 0110
D23.6	110 10111	111010 0110	000101 0110
D24.6	110 11000	110011 0110	001100 0110
D25.6	110 11001	100110 0110	100110 0110
D26.6	110 11010	010110 0110	010110 0110
D27.6	110 11011	110110 0110	001001 0110
D28.6	110 11100	001110 0110	001110 0110
D29.6	110 11101	101110 0110	010001 0110
D30.6	110 11110	011110 0110	100001 0110
D31.6	110 11111	101011 0110	010100 0110
D0.7	111 00000	100111 0001	011000 1110
D1.7	111 00001	011101 0001	100010 1110
D2.7	111 00010	101101 0001	010010 1110
D3.7	111 00011	110001 1110	110001 0001
D4.7	111 00100	110101 0001	001010 1110
D5.7	111 00101	101001 1110	101001 0001
D6.7	111 00110	011001 1110	011001 0001
D7.7	111 00111	111000 1110	000111 0001
D8.7	111 01000	111001 0001	000110 1110
D9.7	111 01001	100101 1110	100101 0001
D10.7	111 01010	010101 1110	010101 0001
D11.7	111 01011	110100 1110	110100 1000
D12.7	111 01100	001101 1110	001101 0001
D13.7	111 01101	101100 1110	101100 1000
D14.7	111 01110	011100 1110	011100 1000
D15.7	111 01111	010111 0001	101000 1110
D16.7	111 10000	011011 0001	100100 1110
D17.7	111 10001	100011 0111	100011 0001
D18.7	111 10010	010011 0111	010011 0001
D19.7	111 10011	110010 1110	110010 0001
D20.7	111 10100	001011 0111	001011 0001

**Table C-1: Valid Data Characters (Cont'd)**

<b>Data Byte Name</b>	<b>Bits HGF EDCBA</b>	<b>Current RD – abcdei fghj</b>	<b>Current RD + abcdei fghj</b>
D21.7	111 10101	101010 1110	101010 0001
D22.7	111 10110	011010 1110	011010 0001
D23.7	111 10111	111010 0001	000101 1110
D24.7	111 11000	110011 0001	001100 1110
D25.7	111 11001	100110 1110	100110 0001
D26.7	111 11010	010110 1110	010110 0001
D27.7	111 11011	110110 0001	001001 1110
D28.7	111 11100	001110 1110	001110 0001
D29.7	111 11101	101110 0001	010001 1110
D30.7	111 11110	011110 0001	100001 1110
D31.7	111 11111	101011 0001	010100 1110

**Table C-2: Valid Control K Characters**

<b>Special Code Name</b>	<b>Bits HGF EDCBA</b>	<b>Current RD – abcdei fghj</b>	<b>Current RD + abcdei fghj</b>
K28.0	000 11100	001111 0100	110000 1011
K28.1	001 11100	001111 1001	110000 0110
K28.2	010 11100	001111 0101	110000 1010
K28.3	011 11100	001111 0011	110000 1100
K28.4	100 11100	001111 0010	110000 1101
K28.5	101 11100	001111 1010	110000 0101
K28.6	110 11100	001111 0110	110000 1001
K28.7 <sup>(1)</sup>	111 11100	001111 1000	110000 0111
K23.7	111 10111	111010 1000	000101 0111
K27.7	111 11011	110110 1000	001001 0111
K29.7	111 11101	101110 1000	010001 0111
K30.7	111 11110	011110 1000	100001 0111

**Notes:**

- Used for testing and characterization only.



# *DRP Address Map of the GTP Transceiver*

---

**Table D-1** lists the DRP map of the GTPE2\_COMMON primitive sorted by address.

**Note:** The reserved bits should NOT be modified. Attributes that are not described explicitly are set automatically by the 7 Series FPGAs Transceivers Wizard. These attributes must be left at their defaults, except for use cases that explicitly request different values.

*Table D-1: DRP Map of GTPE2\_COMMON Primitive*

DRP Address (Hex)	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Binary Encoding
0002	15:0	R/W	PLL0_CFG	15:0	0-65535	0-65535
0003	10:0	R/W	PLL0_CFG	26:16	0-2047	0-2047
0004	13:9	R/W	PLL0_REFCLK_DIV	4:0	1	16
					2	0
0004	7	R/W	PLL0_FBDIV_45	0	4	0
					5	1
0004	5:0	R/W	PLL0_FBDIV	5:0	1	16
					2	0
					3	1
					4	2
					5	3
0005	8:0	R/W	PLL0_LOCK_CFG	8:0	0-511	0-511
0006	15:0	R/W	PLL0_INIT_CFG	15:0	0-65535	0-65535
0007	7:0	R/W	PLL0_INIT_CFG	23:16	0-255	0-255
000A	15:0	R/W	RSVD_ATTR0	15:0	0-65535	0-65535
000F	1	R/W	PLL1_DMON_CFG	0	0-1	0-1
000F	0	R/W	PLL0_DM0N_CFG	0	0-1	0-1
0011	15:0	R/W	COMMON_CFG	15:0	0-65535	0-65535
0012	15:0	R/W	COMMON_CFG	31:16	0-65535	0-65535
0013	7:0	R/W	PLL_CLKOUT_CFG	7:0	0-255	0-255

Table D-1: DRP Map of GTPE2\_COMMON Primitive (Cont'd)

DRP Address (Hex)	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Binary Encoding
0019	15:0	R/W	BIAS_CFG	15:0	0-65535	0-65535
001A	15:0	R/W	BIAS_CFG	31:16	0-65535	0-65535
001B	15:0	R/W	BIAS_CFG	47:32	0-65535	0-65535
001C	15:0	R/W	BIAS_CFG	63:48	0-65535	0-65535
0024	15:0	R/W	RSVD_ATTR1	15:0	0-65535	0-65535
0028	15:0	R/W	PLL1_INIT_CFG	15:0	0-65535	0-65535
0029	7:0	R/W	PLL1_INIT_CFG	23:16	0-255	0-255
002A	8:0	R/W	PLL1_LOCK_CFG	8:0	0-511	0-511
002B	13:9	R/W	PLL1_REFCLK_DIV	4:0	1	16
					2	0
002B	7	R/W	PLL1_FBDIV_45	0	4	0
					5	1
002B	5:0	R/W	PLL1_FBDIV	5:0	1	16
					2	0
					3	1
					4	2
					5	3
002C	15:0	R/W	PLL1_CFG	15:0	0-65535	0-65535
002D	10:0	R/W	PLL1_CFG	26:16	0-2047	0-2047

**Table D-2** lists the DRP map of the GTPE2\_CHANNEL primitive sorted by address.

**Note:** The reserved bits should NOT be modified. Attributes that are not described explicitly are set automatically by the 7 Series FPGAs Transceivers Wizard. These attributes must be left at their defaults, except for use cases that explicitly request different values.

**Table D-2: DRP Map of GTPE2\_CHANNEL Primitive**

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0000	15	R/W	ACJTAG_RESET	0	0-1	0-1
0000	14	R/W	ACJTAG_DEBUG_MODE	0	0-1	0-1
0000	13	R/W	ACJTAG_MODE	0	0-1	0-1
0000	1	R/W	UCODEER_CLR	0	0-1	0-1
000C	15:11	R/W	RXBUFRESET_TIME	4:0	0-31	0-31
000D	14:10	R/W	RXCDRPHRESET_TIME	4:0	0-31	0-31
000D	9:5	R/W	RXCDFREQRESET_TIME	4:0	0-31	0-31
000D	4:0	R/W	RXPMARESET_TIME	4:0	0-31	0-31
000E	11:7	R/W	RXPCSRESET_TIME	4:0	0-31	0-31
000E	6:0	R/W	RXLPMRESET_TIME	6:0	0-127	0-127
000F	11:7	R/W	RXISCANRESET_TIME	4:0	0-31	0-31
0010	15	R/W	RXSYNC_OVRD	0	0-1	0-1
0010	14	R/W	TXSYNC_OVRD	0	0-1	0-1
0010	13	R/W	RXSYNC_SKIP_DA	0	0-1	0-1
0010	12	R/W	TXSYNC_SKIP_DA	0	0-1	0-1
0010	11	R/W	TXSYNC_MULTILANE	0	0-1	0-1
0010	10	R/W	RXSYNC_MULTILANE	0	0-1	0-1
0010	9:5	R/W	TXPCSRESET_TIME	4:0	0-31	0-31
0010	4:0	R/W	TXPMARESET_TIME	4:0	0-31	0-31
0011	14	R/W	RX_XCLK_SEL	0	RXREC	0
					RXUSR	1
0011	13:11	R/W	RX_DATA_WIDTH	2:0	16	2
					20	3
					32	4
					40	5
0011	10:6	R/W	RX_CLK25_DIV	4:0	1	0
					2	1
					3	2
					4	3

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0011 <i>(Cont'd)</i>	10:6	R/W	RX_CLK25_DIV	4:0	5	4
					6	5
					7	6
					8	7
					9	8
					10	9
					11	10
					12	11
					13	12
					14	13
					15	14
					16	15
					17	16
					18	17
					19	18
					20	19
					21	20
					22	21
					23	22
					24	23
					25	24
					26	25
					27	26
					28	27
					29	28
					30	29
					31	30
					32	31
0011	5:4	R/W	RX_CM_SEL	1:0	0-3	0-3
0011	0	R/W	RXPRBS_ERR_LOOPBACK	0	0-1	0-1
0012	15:12	R/W	SATA_BURST_SEQ_LEN	3:0	0-15	0-15
0012	11:10	R/W	OUTREFCLK_SEL_INV	1:0	0-3	0-3

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0012	9:7	R/W	SATA_BURST_VAL	2:0	0-7	0-7
0012	6:0	R/W	RXOOB_CFG	6:0	0-127	0-127
0013	14:9	R/W	SAS_MIN_COM	5:0	1	1
					2	2
					3	3
					4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0013 <i>(Cont'd)</i>	14:9	R/W	SAS_MIN_COM	5:0	31	31
					32	32
					33	33
					34	34
					35	35
					36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0013	8:3	R/W	SATA_MIN_BURST	5:0	1	1
					2	2
					3	3
					4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0013 <i>(Cont'd)</i>	8:3	R/W	SATA_MIN_BURST	5:0	33	33
					34	34
					35	35
					36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
0013	2:0	R/W	SATA_EIDLE_VAL	2:0	0-7	0-7
0014	11:6	R/W	SATA_MIN_WAKE	5:0	1	1
					2	2

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0014 ( <i>Cont'd</i> )	11:6	R/W	SATA_MIN_WAKE	5:0	3	3
					4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0014 <i>(Cont'd)</i>	11:6	R/W	SATA_MIN_WAKE	5:0	35	35
					36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63
0014	5:0	R/W	SATA_MIN_INIT	5:0	1	1
					2	2
					3	3

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0014 <i>(Cont'd)</i>	5:0	R/W	SATA_MIN_INIT	5:0	4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34
					35	35

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0014 <i>(Cont'd)</i>	5:0	R/W	SATA_MIN_INIT	5:0	36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63
0015	12:6	R/W	SAS_MAX_COM	6:0	1	1
					2	2
					3	3
					4	4

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0015 <i>(Cont'd)</i>	12:6	R/W	SAS_MAX_COM	6:0	5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34
					35	35

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0015 <i>(Cont'd)</i>	12:6	R/W	SAS_MAX_COM	6:0	36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66	

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0015 ( <i>Cont'd</i> )	12:6	R/W	SAS_MAX_COM	6:0	67	67
					68	68
					69	69
					70	70
					71	71
					71	71
					73	73
					74	74
					75	75
					76	76
					77	77
					78	78
					79	79
					80	80
					81	81
					82	82
					83	83
					84	84
					85	85
					86	86
					87	87
					88	88
					89	89
					90	90
					91	91
					92	92
					93	93
					94	94
					95	95
					96	96
					97	97
					98	98

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0015 <i>(Cont'd)</i>	12:6	R/W	SAS_MAX_COM	6:0	99	99
					100	100
					101	101
					102	102
					103	103
					104	104
					105	105
					106	106
					107	107
					108	108
					109	109
					110	110
					111	111
					112	112
					113	113
					114	114
					115	115
					116	116
					117	117
					118	118
					119	119
					120	120
					121	121
					122	122
					123	123
					124	124
					125	125
					126	126
					127	127
0015	5:0	R/W	SATA_MAX_BURST	5:0	1	1
					2	2
					3	3

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0015 ( <i>Cont'd</i> )	5:0	R/W	SATA_MAX_BURST	5:0	4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34
					35	35

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0015 <i>(Cont'd)</i>	5:0	R/W	SATA_MAX_BURST	5:0	36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63
0016	11:6	R/W	SATA_MAX_WAKE	5:0	1	1
					2	2
					3	3
					4	4

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0016 ( <i>Cont'd</i> )	11:6	R/W	SATA_MAX_WAKE	5:0	5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34
					35	35
					36	36

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0016 <i>(Cont'd)</i>	11:6	R/W	SATA_MAX_WAKE	5:0	37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63
0016	5:0	R/W	SATA_MAX_INIT	5:0	1	1
					2	2
					3	3
					4	4
					5	5

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0016 <i>(Cont'd)</i>	5:0	R/W	SATA_MAX_INIT	5:0	6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37	6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0016 <i>(Cont'd)</i>	5:0	R/W	SATA_MAX_INIT	5:0	38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63
0017	15:11	R/W	RXOSCALRESET_TIMEOUT	4:0	0-31	0-31
0017	10:6	R/W	RXOSCALRESET_TIME	4:0	0-31	0-31
0018	7:0	R/W	TRANS_TIME_RATE	7:0	0-255	0-255
0019	15	R/W	PMA_LOOPBACK_CFG	0	0-1	0-1
0019	12	R/W	TX_PREDRIVER_MODE	0	0-1	0-1
0019	11:9	R/W	TX_EIDLE_DEASSERT_DELAY	2:0	0-7	0-7

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0019	8:6	R/W	TX_IDLE_ASSERT_DELAY	2:0	0-7	0-7
0019	5	R/W	TX_LOOPBACK_DRIVE_HIZ	0	FALSE	0
					TRUE	1
0019	4:0	R/W	TX_DRIVE_MODE	0	DIRECT	0
					PIPE	1
001A	15:8	R/W	PD_TRANS_TIME_TO_P2	7:0	0-255	0-255
001A	7:0	R/W	PD_TRANS_TIME_NONE_P2	7:0	0-255	0-255
001B	12:1	R/W	PD_TRANS_TIME_FROM_P2	11:0	0-4095	0-4095
001B	0	R/W	PCS_PCIE_EN	0	FALSE	0
					TRUE	1
001C	15	R/W	TXBUF_RESET_ON_RATE_CHANGE	0	FALSE	0
					TRUE	1
001C	14	R/W	TXBUF_EN	0	FALSE	0
					TRUE	1
001C	5	R/W	TXGEARBOX_EN	0	FALSE	0
					TRUE	1
001C	2:0	R/W	GEARBOX_MODE	2:0	0-7	0-7
001E	14	R/W	RXLPM_HOLD_DURING_EIDLE	0	0-1	0-1
0024	12:0	R/W	RX_OS_CFG	12:0	0-8191	0-8191
002A	15:14	R/W	RXLPM_LF_CFG	17:16	0-3	0-3
002A	13:0	R/W	RXLPM_HF_CFG	13:0	0-16383	0-16383
002B	15:0	R/W	RXLPM_LF_CFG	15:0	0-65535	0-65535
002C	15:0	R/W	ES_QUALIFIER	15:0	0-65535	0-65535
002D	15:0	R/W	ES_QUALIFIER	31:16	0-65535	0-65535
002E	15:0	R/W	ES_QUALIFIER	47:32	0-65535	0-65535
002F	15:0	R/W	ES_QUALIFIER	63:48	0-65535	0-65535
0030	15:0	R/W	ES_QUALIFIER	79:64	0-65535	0-65535
0031	15:0	R/W	ES_QUAL_MASK	15:0	0-65535	0-65535
0032	15:0	R/W	ES_QUAL_MASK	31:16	0-65535	0-65535
0033	15:0	R/W	ES_QUAL_MASK	47:32	0-65535	0-65535
0034	15:0	R/W	ES_QUAL_MASK	63:48	0-65535	0-65535
0035	15:0	R/W	ES_QUAL_MASK	79:64	0-65535	0-65535

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0036	15:0	R/W	ES_SDATA_MASK	15:0	0-65535	0-65535
0037	15:0	R/W	ES_SDATA_MASK	31:16	0-65535	0-65535
0038	15:0	R/W	ES_SDATA_MASK	47:32	0-65535	0-65535
0039	15:0	R/W	ES_SDATA_MASK	63:48	0-65535	0-65535
003A	15:0	R/W	ES_SDATA_MASK	79:64	0-65535	0-65535
003B	15:11	R/W	ES_PRESCALE	4:0	0-31	0-31
003B	8:0	R/W	ES_VERT_OFFSET	8:0	0-511	0-511
003C	11:0	R/W	ES_HORZ_OFFSET	11:0	0-4095	0-4095
003D	15	R/W	RX_DISPERR_SEQ_MATCH	0	FALSE	0
					TRUE	1
003D	14	R/W	DEC_PCOMMA_DETECT	0	FALSE	0
					TRUE	1
003D	13	R/W	DEC_MCOMMA_DETECT	0	FALSE	0
					TRUE	1
003D	12	R/W	DEC_VALID_COMMA_ONLY	0	FALSE	0
					TRUE	1
003D	9	R/W	ES_ERRDET_EN	0	FALSE	0
					TRUE	1
003D	8	R/W	ES_EYE_SCAN_EN	0	FALSE	0
					TRUE	1
003D	5:0	R/W	ES_CONTROL	5:0	0-63	0-63
003E	9:0	R/W	ALIGN_COMMMA_ENABLE	9:0	0-1023	0-1023
003F	9:0	R/W	ALIGN_MCOMMA_VALUE	9:0	0-1023	0-1023
0040	15:14	R/W	RXSLIDE_MODE	1:0	OFF	0
					AUTO	1
					PCS	2
					PMA	3
0040	9:0	R/W	ALIGN_PCOMMA_VALUE	9:0	0-1023	0-1023
0041	14:13	R/W	ALIGN_COMMMA_WORD	1:0	1	1
					2	2
0041	12:8	R/W	RX_SIG_VALID_DLY	4:0	1	0
					2	1

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0041 <i>(Cont'd)</i>	12:8	R/W	RX_SIG_VALID_DLY	4:0	3	2
					4	3
					5	4
					6	5
					7	6
					8	7
					9	8
					10	9
					11	10
					12	11
					13	12
					14	13
					15	14
					16	15
					17	16
					18	17
					19	18
					20	19
					21	20
					22	21
					23	22
					24	23
					25	24
					26	25
					27	26
					28	27
					29	28
					30	29
					31	30
					32	31
0041	7	R/W	ALIGN_PCOMMA_DET	0	FALSE	0
					TRUE	1

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0041	6	R/W	ALIGN_MCOMMA_DET	0	FALSE	0
					TRUE	1
0041	5	R/W	SHOW_REALIGN_COMMA	0	FALSE	0
					TRUE	1
0041	4	R/W	ALIGN_COMM_DOUBLE	0	FALSE	0
					TRUE	1
0041	3:0	R/W	RXSLIDE_AUTO_WAIT	3:0	0	0
					1	1
					2	2
					3	3
					4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
0044	14	R/W	CLK_CORRECT_USE	0	FALSE	0
					TRUE	1
0044	13:10	R/W	CLK_COR_SEQ_1_ENABLE	3:0	0-15	0-15
0044	9:0	R/W	CLK_COR_SEQ_1_1	9:0	0-1023	0-1023
0045	15:10	R/W	CLK_COR_MAX_LAT	5:0	6	6
					7	7
					8	8

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0045 ( <i>Cont'd</i> )	15:10	R/W	CLK_COR_MAX_LAT	5:0	9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34
					35	35
					36	36
					37	37
					38	38
					39	39

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0045 <i>(Cont'd)</i>	15:10	R/W	CLK_COR_MAX_LAT	5:0	40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
0045	9:0	R/W	CLK_COR_SEQ_1_2	9:0	0-1023	0-1023
0046	15:10	R/W	CLK_COR_MIN_LAT	5:0	4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0046 ( <i>Cont'd</i> )	15:10	R/W	CLK_COR_MIN_LAT	5:0	13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34
					35	35
					36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0046 ( <i>Cont'd</i> )	15:10	R/W	CLK_COR_MIN_LAT	5:0	45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60	45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
0046	9:0	R/W	CLK_COR_SEQ_1_3	9:0	0-1023	0-1023
0047	14:10	R/W	CLK_COR_REPEAT_WAIT	4:0	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0047 ( <i>Cont'd</i> )	14:10	R/W	CLK_COR_REPEAT_WAIT	4:0	15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
0047	9:0	R/W	CLK_COR_SEQ_1_4	9:0	0-1023	0-1023
0048	14	R/W	CLK_COR_SEQ_2_USE	0	FALSE	0
					TRUE	1
0048	13:10	R/W	CLK_COR_SEQ_2_ENABLE	3:0	0-15	0-15
0048	9:0	R/W	CLK_COR_SEQ_2_1	9:0	0-1023	0-1023
0049	14	R/W	CLK_COR_KEEP_IDLE	0	FALSE	0
					TRUE	1
0049	12	R/W	CLK_COR_PRECEDENCE	0	FALSE	0
					TRUE	1
0049	11:10	R/W	CLK_COR_SEQ_LEN	1:0	1	0
					2	1
					3	2
					4	3
0049	9:0	R/W	CLK_COR_SEQ_2_2	9:0	0-1023	0-1023
004A	9:0	R/W	CLK_COR_SEQ_2_3	9:0	0-1023	0-1023

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
004B	15	R/W	RXGEARBOX_EN	0	FALSE	0
					TRUE	1
004B	9:0	R/W	CLK_COR_SEQ_2_4	9:0	0-1023	0-1023
004C	15:12	R/W	CHAN_BOND_SEQ_1_ENABLE	3:0	0-15	0-15
004C	9:0	R/W	CHAN_BOND_SEQ_1_1	9:0	0-1023	0-1023
004D	15:14	R/W	CHAN_BOND_SEQ_LEN	1:0	1	0
					2	1
					3	2
					4	3
004D	9:0	R/W	CHAN_BOND_SEQ_1_2	9:0	0-1023	0-1023
004E	15	R/W	CHAN_BOND_KEEP_ALIGN	0	FALSE	0
					TRUE	1
004E	9:0	R/W	CHAN_BOND_SEQ_1_3	9:0	0-1023	0-1023
004F	9:0	R/W	CHAN_BOND_SEQ_1_4	9:0	0-1023	0-1023
0050	15:12	R/W	CHAN_BOND_SEQ_2_ENABLE	3:0	0-15	0-15
0050	11	R/W	CHAN_BOND_SEQ_2_USE	0	FALSE	0
					TRUE	1
0050	9:0	R/W	CHAN_BOND_SEQ_2_1	9:0	0-1023	0-1023
0051	15:12	R/W	FTS_LANE_DESKEW_CFG	3:0	0-15	0-15
0051	11	R/W	FTS_LANE_DESKEW_EN	0	FALSE	0
					TRUE	1
0051	9:0	R/W	CHAN_BOND_SEQ_2_2	9:0	0-1023	0-1023
0052	15:12	R/W	FTS_DESKEW_SEQ_ENABLE	3:0	0-15	0-15
0052	11	R/W	CBCC_DATA_SOURCE_SEL	0	ENCODED	0
					DECODED	0
0052	9:0	R/W	CHAN_BOND_SEQ_2_3	9:0	0-1023	0-1023
0053	15:12	R/W	CHAN_BOND_MAX_SKEW	3:0	1	1
					2	2
					3	3
					4	4
					5	5
					6	6

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0053 ( <i>Cont'd</i> )	15:12	R/W	CHAN_BOND_MAX_SKEW	3:0	7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
0053	9:0	R/W	CHAN_BOND_SEQ_2_4	9:0	0-1023	0-1023
0054	15:0	R/W	RXDLY_TAP_CFG	15:0	0-65535	0-65535
0055	15:0	R/W	RXDLY_CFG	15:0	0-65535	0-65535
0057	12:8	R/W	RXPH_MONITOR_SEL	4:0	0-31	0-31
0057	5:0	R/W	RX_DDI_SEL	5:0	0-63	0-63
0059	7	R/W	TX_XCLK_SEL	0	TXOUT	0
					TXUSR	1
0059	6	R/W	RXBUF_EN	0	FALSE	0
					TRUE	1
005A	9	R/W	TXOOB_CFG	0	0-1	0-1
005A	8	R/W	LOOPBACK_CFG	0	0-1	0-1
005D	10:8	R/W	TXPI_CFG5	2:0	0-7	0-7
005D	7	R/W	TXPI_CFG4	0	0-1	0-1
005D	6	R/W	TXPI_CFG3	0	0-1	0-1
005D	5:4	R/W	TXPI_CFG2	1:0	0-3	0-3
005D	3:2	R/W	TXPI_CFG1	1:0	0-3	0-3
005D	1:0	R/W	TXPI_CFG0	1:0	0-3	0-3
005E	15:14	R/W	SATA_PLL_CFG	1:0	VCO_3000MHZ	0
					VCO_1500MHZ	1
					VCO_750MHZ	2
0060	15:0	R/W	TXPHDLY_CFG	15:0	0-65535	0-65535
0061	7:0	R/W	TXPHDLY_CFG	23:16	0-255	0-255
0062	15:0	R/W	TXDLY_CFG	15:0	0-65535	0-65535
0063	15:0	R/W	TXDLY_TAP_CFG	15:0	0-65535	0-65535

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
0064	15:0	R/W	TXPH_CFG	15:0	0-65535	0-65535
0065	12:8	R/W	TXPH_MONITOR_SEL	4:0	0-31	0-31
0066	15:0	R/W	RX_BIAS_CFG	15:0	0-65535	0-65535
0068	3	R/W	RXOOB_CLK_CFG	0	PMA	0
					FABRIC	1
0068	1	R/W	TX_CLKMUX_EN	0	0-1	0-1
0068	0	R/W	RX_CLKMUX_EN	0	0-1	0-1
0069	14:0	R/W	TERM_RCAL_CFG	14:0	0-32767	0-32767
006A	15:13	R/W	TERM_RCAL_OVRD	2:0	0-7	0-7
006A	4:0	R/W	TX_CLK25_DIV	4:0	1	0
					2	1
					3	2
					4	3
					5	4
					6	5
					7	6
					8	7
					9	8
					10	9
					11	10
					12	11
					13	12
					14	13
					15	14
					16	15
					17	16
					18	17
					19	18
					20	19
					21	20
					22	21
					23	22

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
006A ( <i>Cont'd</i> )	4:0	R/W	TX_CLK25_DIV	4:0	24	23
					25	24
					26	25
					27	26
					28	27
					29	28
					30	29
					31	30
					32	31
006B	15	R/W	PMA_RSV5	0	0-1	0-1
006B	11:8	R/W	PMA_RSV4	3:0	0-15	0-15
006B	2:0	R/W	TX_DATA_WIDTH	2:0	16	2
					20	3
					32	4
					40	5
006F	15:0	R/W	PCS_RSVD_ATTR	15:0	0-65535	0-65535
0070	15:0	R/W	PCS_RSVD_ATTR	31:16	0-65535	0-65535
0071	15:0	R/W	PCS_RSVD_ATTR	47:32	0-65535	0-65535
0075	14:8	R/W	TX_MARGIN_FULL_1	6:0	0-127	0-127
0075	6:0	R/W	TX_MARGIN_FULL_0	6:0	0-127	0-127
0076	14:8	R/W	TX_MARGIN_FULL_3	6:0	0-127	0-127
0076	6:0	R/W	TX_MARGIN_FULL_2	6:0	0-127	0-127
0077	14:8	R/W	TX_MARGIN_LOW_0	6:0	0-127	0-127
0077	6:0	R/W	TX_MARGIN_FULL_4	6:0	0-127	0-127
0078	14:8	R/W	TX_MARGIN_LOW_2	6:0	0-127	0-127
0078	6:0	R/W	TX_MARGIN_LOW_1	6:0	0-127	0-127
0079	14:8	R/W	TX_MARGIN_LOW_4	6:0	0-127	0-127
0079	6:0	R/W	TX_MARGIN_LOW_3	6:0	0-127	0-127
007A	13:8	R/W	TX_DEEMPH1	5:0	0-63	0-63
007A	5:0	R/W	TX_DEEMPH0	5:0	0-63	0-63
007C	10:8	R/W	TX_RXDETECT_REF	2:0	0-7	0-7
007C	3	R/W	TX_MAINCURSOR_SEL	0	0-1	0-1

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
007C	1:0	R/W	PMA_RSV3	1:0	0-3	0-3
007D	15	R/W	PMA_RSV7	0	0-1	0-1
007D	14	R/W	PMA_RSV6	0	0-1	0-1
007D	13:0	R/W	TX_RXDETECT_CFG	13:0	0-16383	0-16383
007E	15	R/W	CLK_COMMON_SWING	0	0-1	0-1
007E	3:0	R/W	RX_CM_TRIM	3:0	0-15	0-15
0081	4	R/W	RXLPM_CFG1	0	0-1	0-1
0081	3:0	R/W	RXLPM_CFG	3:0	0-15	0-15
0082	15:0	R/W	PMA_RSV2	15:0	0-65535	0-65535
0083	15:0	R/W	PMA_RSV2	31:16	0-65535	0-65535
0086	15:0	R/W	DMONITOR_CFG	15:0	0-65535	0-65535
0087	7:0	R/W	DMONITOR_CFG	23:16	0-255	0-255
0088	15	R/W	RXLPM_BIAS_STARTUP_DISABLE	0	0-1	0-1
0088	14:11	R/W	RXLPM_HF_CFG3	3:0	0-15	0-15
0088	6:4	R/W	TXOUT_DIV	2:0	1	0
					2	1
					4	2
					8	3
0088	2:0	R/W	RXOUT_DIV	2:0	1	0
					2	1
					4	2
					8	3
0089	15:0	R/W	CFOK_CFG	15:0	0-65535	0-65535
008A	15:0	R/W	CFOK_CFG	31:16	0-65535	0-65535
008B	10:0	R/W	CFOK_CFG	42:32	0-2047	0-2047
008C	6:0	R/W	CFOK_CFG3	6:0	0-127	0-127
008D	15:13	R/W	RXPI_CFG0	2:0	0-7	0-7
008D	12	R/W	RXLPM_CM_CFG	0	0-1	0-1
008D	11:10	R/W	CFOK_CFG5	1:0	0-3	0-3
008D	9:5	R/W	RXLPM_LF_CFG2	4:0	0-31	0-31
008D	4:0	R/W	RXLPM_HF_CFG2	4:0	0-31	0-31
008E	15	R/W	RXLPM_IPCM_CFG	0	0-1	0-1

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
008E	14	R/W	RXLPM_INCM_CFG	0	0-1	0-1
008E	13	R/W	CFOK_CFG4	0	0-1	0-1
008E	12:9	R/W	CFOK_CFG6	3:0	0-15	0-15
008E	8:0	R/W	RXLPM_GC_CFG	8:0	0-511	0-511
008F	7:5	R/W	RXLPM_GC_CFG2	2:0	0-7	0-7
008F	4	R/W	RXPI_CFG1	0	0-1	0-1
008F	3	R/W	RXPI_CFG2	0	0-1	0-1
008F	2:0	R/W	RXLPM_OSINT_CFG	2:0	0-7	0-7
0091	15	R/W	ES_CLK_PHASE_SEL	0	0-1	0-1
0091	14	R/W	USE_PCS_CLK_PHASE_SEL	0	0-1	0-1
0091	12:6	R/W	CFOK_CFG2	6:0	0-127	0-127
0092	15:0	R/W	ADAPT_CFG0	15:0	0-65535	0-65535
0093	3:0	R/W	ADAPT_CFG0	19:16	0-15	0-15
0095	7:0	R/W	TXPI_PPM_CFG	7:0	0-255	0-255
0096	5	R/W	TXPI_GREY_SEL	0	0-1	0-1
0096	4	R/W	TXPI_INVSTROBE_SEL	0	0-1	0-1
0096	3	R/W	TXPI_PPMCLK_SEL	0	TXUSRCLK	0
					TXUSRCLK2	1
0096	2:0	R/W	TXPI_SYNFFREQ_PPM	2:0	0-7	0-7
0097	15:0	R/W	TST_RSV	15:0	0-65535	0-65535
0098	15:0	R/W	TST_RSV	31:16	0-65535	0-65535
0099	15:0	R/W	PMA_RSV	15:0	0-65535	0-65535
009A	15:0	R/W	PMA_RSV	31:16	0-65535	0-65535
009B	5:0	R/W	RX_BUFFER_CFG	5:0	0-63	0-63
009C	8	R/W	RXBUF_THRESH_OVRD	0	FALSE	0
					TRUE	1
009C	6	R/W	RXBUF_RESET_ON_EIDLE	0	FALSE	0
					TRUE	1
009C	5:0	R/W	RXBUF_THRESH_UNDFLW	5:0	0	0
					1	1
					2	2
					3	3

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
009C <i>(Cont'd)</i>	5:0	R/W	RXBUF_THRESH_UNDFLW	5:0	4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31
					32	32
					33	33
					34	34
					35	35

**Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)**

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
009C <i>(Cont'd)</i>	5:0	R/W	RXBUF_THRESH_UNDFLW	5:0	36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63
009D	15:12	R/W	RXBUF_EIDLE_HI_CNT	3:0	0-15	0-15
009D	11:8	R/W	RXBUF_EIDLE_LO_CNT	3:0	0-15	0-15
009D	7	R/W	RXBUF_ADDR_MODE	0	FULL	0
					FAST	1

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
009D	6:1	R/W	RXBUFF_THRESH_OVFLW	5:0	0	0
					1	1
					2	2
					3	3
					4	4
					5	5
					6	6
					7	7
					8	8
					9	9
					10	10
					11	11
					12	12
					13	13
					14	14
					15	15
					16	16
					17	17
					18	18
					19	19
					20	20
					21	21
					22	22
					23	23
					24	24
					25	25
					26	26
					27	27
					28	28
					29	29
					30	30
					31	31

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (*Cont'd*)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
009D <i>(Cont'd)</i>	6:1	R/W	RXBUFF_THRESH_OVFLW	5:0	32	32
					33	33
					34	34
					35	35
					36	36
					37	37
					38	38
					39	39
					40	40
					41	41
					42	42
					43	43
					44	44
					45	45
					46	46
					47	47
					48	48
					49	49
					50	50
					51	51
					52	52
					53	53
					54	54
					55	55
					56	56
					57	57
					58	58
					59	59
					60	60
					61	61
					62	62
					63	63

Table D-2: DRP Map of GTPE2\_CHANNEL Primitive (Cont'd)

DRP Address	DRP Bits	R/W	Attribute Name	Attribute Bits	Attribute Encoding	DRP Encoding
009D	0	R/W	RX_DEFER_RESET_BUF_EN	0	FALSE	0
					TRUE	1
009E	2	R/W	RXBUF_RESET_ON_COMMALIGN	0	FALSE	0
					TRUE	1
009E	1	R/W	RXBUF_RESET_ON_RATE_CHANGE	0	FALSE	0
					TRUE	1
009E	0	R/W	RXBUF_RESET_ON_CB_CHANGE	0	FALSE	0
					TRUE	1
009F	8:0	R/W	TXDLY_LCFG	8:0	0-511	0-511
00A0	8:0	R/W	RXDLY_LCFG	8:0	0-511	0-511
00A1	15:0	R/W	RXPH_CFG	15:0	0-65535	0-65535
00A2	7:0	R/W	RXPH_CFG	23:16	0-255	0-255
00A3	15:0	R/W	RXPHDLY_CFG	15:0	0-65535	0-65535
00A4	7:0	R/W	RXPHDLY_CFG	23:16	0-255	0-255
00A5	13:0	R/W	RX_DEBUG_CFG	13:0	0-16383	0-16383
00A6	9:0	R/W	ES_PMA_CFG	9:0	0-1023	0-1023
00A7	13	R/W	RXCDR_PH_RESET_ON_EIDLE	0	0-1	0-1
00A7	12	R/W	RXCDR_FR_RESET_ON_EIDLE	0	0-1	0-1
00A7	11	R/W	RXCDR_HOLD_DURING_EIDLE	0	0-1	0-1
00A7	5:0	R/W	RXCDR_LOCK_CFG	5:0	0-63	0-63
00A8	15:0	R/W	RXCDR_CFG	15:0	0-65535	0-65535
00A9	15:0	R/W	RXCDR_CFG	31:16	0-65535	0-65535
00AA	15:0	R/W	RXCDR_CFG	47:32	0-65535	0-65535
00AB	15:0	R/W	RXCDR_CFG	63:48	0-65535	0-65535
00AC	15:0	R/W	RXCDR_CFG	79:64	0-65535	0-65535
00AD	2:0	R/W	RXCDR_CFG	82:80	0-7	0-7